



SPACEOPS
9 2

PROCEEDINGS
OF THE
SECOND INTERNATIONAL SYMPOSIUM
ON GROUND DATA SYSTEMS
FOR SPACE MISSION OPERATIONS

NOVEMBER 16-20, 1992
PASADENA, CALIFORNIA, USA

MARCH 1, 1993
JPL PUBLICATION 93-5

N94-23832
--THRU--
N94-23968
Unclass

G3/66 0182640

(NASA-CR-194486) SPACEOPS 1992:
PROCEEDINGS OF THE SECOND
INTERNATIONAL SYMPOSIUM ON GROUND
DATA SYSTEMS FOR SPACE MISSION
OPERATIONS (JPL) 869 p

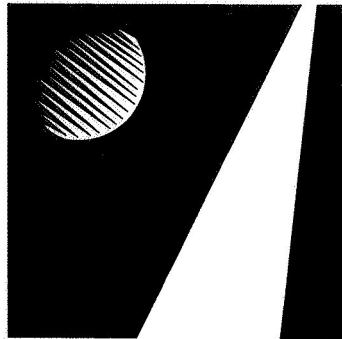
ABSTRACT

The Second International Symposium on Ground Data Systems for Space Mission Operations (SPACEOPS 92), hosted by the Jet Propulsion Laboratory (JPL), was held November 16–20, 1992, in Pasadena, California, USA. More than 400 people from 14 countries attended. This symposium followed the First International Symposium on Ground Data Systems for Spacecraft Control in June 1990, sponsored by the European Space Agency and the European Space Operations Centre.

The Second International Symposium featured 135 oral presentations in these 12 categories:

- Future Missions and Operations
- System-Level Architectures
- Mission-Specific Systems
- Mission and Science Planning and Sequencing
- Mission Control
- Operations Automation and Emerging Technologies
- Data Acquisition
- Navigation
- Operations Support Services
- Engineering Data Analysis of Space Vehicle and Ground Systems
- Telemetry Processing, Mission Data Management, and Data Archiving
- Operations Management

Topics focused on improvements in the productivity, effectiveness, efficiency, and quality of mission operations, ground systems, and data acquisition. New methods, techniques, human systems, and technology were discussed. Also emphasized were accomplishments in the management of human factors; use of information systems to improve data retrieval, reporting, and archiving; design and implementation of logistics support for mission operations; and the use of telescience and teleoperations.



SPACEOPS

9 2

PROCEEDINGS

OF THE

SECOND INTERNATIONAL SYMPOSIUM

ON GROUND DATA SYSTEMS

FOR SPACE MISSION OPERATIONS

NOVEMBER 16–20, 1992

PASADENA, CALIFORNIA, USA

MARCH 1, 1993

JPL PUBLICATION 93-5

FOREWORD

During the four days of SPACEOps 92, more than 400 people from 14 countries heard 135 presentations, as well as keynote, plenary, and panel talks by individuals from throughout the world. The papers in these Proceedings describe a wide range of ideas and experiences in our field as seen from the perspectives of national space programs and industry.

SPACEOps 92 highlighted the challenges that will shape space mission operations in the future:

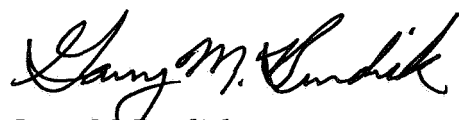
- Working together in cooperative, multinational projects.
- Becoming more cost efficient.
- Using new technologies.
- Developing and implementing new approaches.

The positive response to SPACEOps 92 has demonstrated that the time has arrived for members of the space mission operations community to regularly meet and share their knowledge in a cooperative, multinational environment. The Jet Propulsion Laboratory (JPL) is proud to follow the European Space Agency (ESA) and the European Space Operations Centre (ESOC) as host and sponsor. We, like other participating organizations, benefit from the shared knowledge and associations of this event.

Planning has already begun for SPACEOps 94 and for other future symposia. Those of us who organized this symposium look forward to continuing the associations begun here and to sharing a future of successful space mission programs.



Norman R. Haynes
Chairperson



Garry M. Burdick
Vice Chairperson

CONTENTS

Keynote Speech	1
<i>Strategies in Transition</i>	
Alphonso V. Diaz	
Plenary Speeches	
<i>Commonality of Flight Control Systems for Support of</i>	
<i>European Telecommunications Missions</i>	7
Kurt Debatin	
<i>NASDA's View of Ground Control in Mission Operations</i>	15
Satoshi Tateno	
<i>Future Trends in NASA Space Operations</i>	21
Charles F. Fuechsel	
<i>The Space Operations at CNES</i>	25
Roland Ivarnez	
Session A	29
Future Missions and Operations	
Session B	87
System-Level Architectures	
Session C	155
Mission-Specific Systems	
Session D	215
Mission and Science Planning and Sequencing	
Session E	297
Mission Control	
Session F	399
Operations Automation and Emerging Technologies	
Session G	491
Data Acquisition	
Session H	573
Navigation	
Session I	641
Operations Support Services	
Session J	717
Engineering Data Analysis of Space Vehicle and Ground Systems	
Session K	763
Telemetry Processing, Mission Data Management, and Data Archiving	
Session L	841
Operations Management	
Keyword Index	905
Author Index	909
Acknowledgments	911

KEYNOTE SPEECH

STRATEGIES IN TRANSITION N94-23833

ALPHONSO V. DIAZ
Deputy Associate Administrator
Office of Space Science and Applications
National Aeronautics and Space Administration

I'm very pleased to be here. As experts from around the world in mission operations and systems, the task facing you here at the Second International Symposium — to exchange information and ideas, share technology advancements, and discuss ways to increase efficiency — has never been more important. One reason is that this year has been the most vigorous for space science in decades, with more missions launched than at any time in over 20 years, and many with great international cooperation. A second reason is that a new vision has emerged within the Office of Space Science and Applications (OSSA), and within the agency as a whole, for how to design missions to be responsive to the changing budget environment of the 1990s. I thought that it would be helpful to open this symposium by giving you a context for your discussions, some sense of why and how the vision has changed, and what strategy OSSA is implementing to achieve these new objectives.

Let me begin by providing some sense of how the environment in which we do business has changed. For most of the 1980s, the National Aeronautics and Space Administration (NASA) enjoyed very healthy budget growth. Between 1983 and 1992, NASA's budget doubled — from \$7 to \$14 billion. Over the last few years, however, we began to feel the squeeze of competition for constrained Federal resources as a bad Federal funding environment worsened. The 1991 budget agreement — which established walls between domestic discretionary, defense discretionary, and foreign assistance accounts — effectively prohibited NASA from sharing in any peace dividend. It also placed NASA in direct competition with other domestic programs — such as aid to American veterans or housing programs — for

very limited resources. Fiscal years 1992 and 1993 were very difficult budget years. Even Mission Operations and Data Analysis (MO&DA), which is a budget category on the rise with all the missions we are flying, suffered as part of the exercise of balancing the books. Nineteen ninety-four and 1995 will clearly bring more of the same, and in MO&DA, as well as in program development, there is a need for finding efficiencies in the near-term and making investments for more cost-effective operations in the long term.

Prior to this new environment, OSSA's strategic planning activities assumed budget growth of more than 10 percent each year. As our Congressional committees made it clear that the rate of growth that nearly doubled OSSA's budget in five years would not continue, the Space Science and Applications Advisory Committee (SSAAC), as part of its activity of reviewing OSSA's strategic plan in summer 1991, evaluated what changes would be required to keep expected growth in line with reality. Table 1 provides an overview of OSSA's 1991 strategic plan core science program.

While SSAAC reaffirmed the importance of most missions already in development, such as Comet Rendezvous Asteroid Flyby (CRAF), Cassini, the Advanced X-Ray Astrophysics Facility (AXAF), and the Earth Observing System (EOS), the committee made the far-reaching recommendation to emphasize smaller missions, with more frequent access to space and a greater role for the research community external to NASA. The Orbiting Solar Laboratory and Space Infrared Telescope Facility were deferred. SSAAC created the intermediate category of missions to characterize the new leaner and more focused class of science missions. It also established two mission queues — one for intermediate and flagship missions and one for small missions — to further insulate the small missions from the effects of shortfalls in funding.

Table 1. Core Science Program: 1991 OSSA Strategic Plan.

Year	Ongoing Program	Major and Moderate Missions	Small Missions	Space Station Freedom Utilization	Research Base Enhancements
1989	Research and analysis	Advanced X-Ray Astrophysics Facility	Scout-class Explorers	Microgravity facilities 1.8-m centrifuge Attached payloads Earth Observing System payload definition	SETI Microwave Observing Project CRAF/Cassini advanced technology development Supernova 1987A suborbital observation ER-2 purchase
1990	Mission operations and data analysis	CRAF/Cassini	Total Ozone Mapping Spectrometer	Space Biology Initiative definition Earth Observing System payload definition	Research and analysis; missions operations and data analysis corrections
1991	Aerospace medicine Flight projects	Earth Observing System	Earth Probes	Space Biology Initiative Biomedical monitoring and countermeasures	
1992	Spacelabs and other carriers		Earth Probes augmentation		Resources to augment research community Data Revitalization Initiative Studies of mesosphere and lower thermosphere
1993 through 1997		Orbiting Solar Laboratory Space Infrared Telescope Facility Lunar Observer Gravity Probe-B Solar Probe	Microgravity fundamental science	Small and rapid-response payloads	Stratospheric Observatory for Infrared Astronomy Focused research and analysis; suborbital, advanced technology development; data systems enhancements

Even as SSAAC and OSSA tried in late 1991 to flesh out a new strategy, the budget environment worsened, leading to a period of further retrenchment. The environment of constraint led to a number of very difficult decisions for OSSA as well as major restructuring of a few key programs in development. As you know, NASA was forced to recommend the termination of the CRAF mission. The decision was based on the savings that would be attained over the life of the program, particularly in the peak years of 1994, 1995, and in the outyears. The termination of the Magellan mission at the end of 1993 was another difficult decision imposed on OSSA by the environment of constraint.

In addition to these specific program decisions, we had to look at the overall space science and applications program, restructuring the most expensive and complex projects to bring down costs and ensure their place in the mission queue of the future.

First, after CRAF was canceled, we looked very closely at the Cassini program to see how it could be made more secure for future funding cycles. We were successful in rescoping the program to reduce development costs as well as total project costs, while maintaining a world-class science program. The mission now relies on a simpler, lighter spacecraft with body-fixed instruments. The launch schedule has not changed. In fact, now that spacecraft weight has been reduced, the launch strategy is more resilient.

AXAF is another good example of how we are restructuring the largest missions to be leaner and more focused. While we preserved virtually intact the science mission for AXAF, we reduced mission complexity and restructured the program into two smaller, complementary spacecraft, with comparable imaging science and better observing efficiency than the baseline mission, and with lighter, simpler, and less expensive mirrors to be used for spectroscopy. Lighter, simpler, and less expensive — this is the theme of the new environment. The result will be a mission that will likely fly earlier than the original facility, and excellent science will be returned while funding requirements, particularly for peak years, are reduced.

Restructuring was not limited to the traditional space sciences. EOS is the principal element of NASA's Mission to Planet Earth and NASA's contribution to the U.S. Global Change Research Program. It carried an original price tag of \$17 billion through the year 2000 and became another target of restructuring. NASA is going to realize substantial savings by using a common spacecraft design after the first spacecraft, by relying on greater international cooperation, and by pursuing other efficiencies in engineering and analysis. The total development cost has been reduced to \$8 billion. The original science objectives and schedule are intact.

In the 1993 budget, OSSA's efforts met with success — EOS and Cassini were fully funded, and AXAF nearly so. Development costs have been reduced substantially in these programs.

Regardless of the savings in some of these programs in development, the challenge of reducing operations costs for all OSSA missions, existing projects as well as new, remains before us. Recently, we appointed a team to study ways of decreasing costs in mission operations by up to 15 percent in five years, an exercise similar to that conducted by the Space Shuttle people to decrease costs by 3 percent, 6 percent, and then 9 percent within three years. The OSSA team looked at individual programs to see how operations management could be made more efficient. Many cost-cutting approaches were identified, and the team, now in the implementation phase, is beginning to put the first of them into effect.

All the hard work in the last year or two to restructure some of OSSA's largest programs in development and to improve efficiency for those

in operation is part of OSSA's effort to free funds for more frequent space science missions in the future. Instead of more great observatories, we are looking toward a new vision encompassing a level of great activity through small, frequent missions. What we've learned over the last several budget cycles is that in this new environment, we need a highly flexible program that fits into a continuous, smooth level of effort, without a huge bell curve of funding requirements over peak years. There is excellent science return to be gained by this approach in that it will allow for more rapid development, more frequent flight opportunities, and more focused scientific objectives. It will also allow continued access to deep space, which appeared to be increasingly threatened when large programs met funding constraints.

We are now hard at work on implementing the strategy for attaining this vision. The strategy we developed was to lower costs by reducing size and complexity through new technology, while at the same time making progress in space science. The strategy comprises two interwoven parts: the flight program strategy of each of the science disciplines and OSSA's new-technology strategy.

The flight plan component of the strategy is that every science discipline will have a flexible, low-cost set of missions to carry out its science objectives. This set of quicker, cheaper missions serves several purposes: to increase the flight rate to fill the data gaps between larger missions; to increase university and industry participation in the science programs; to provide opportunities for collaboration with other agencies and with our international partners; and to allow for rapid response to new science opportunities that emerge.

NASA has a long history of success with smaller missions, and in fact, the new strategy builds on many programs already in existence in OSSA. The goal is to expand the small-mission programs already in place and to initiate new ones using those successful models. Table 2 shows OSSA's small-mission programmatic strategy.

The Small-Class Explorers (SMEX) program is an excellent example of a program already in place that fits with the new strategy. SMEX physics and astronomy payloads are modestly sized, modest-capability payloads of up to 500 pounds. They provide the astrophysics and space physics programs with a quick-reaction research capability and frequent launch opportunities, and the relatively short development time for SMEX missions allows for the exploration of new science areas

Table 2. OSSA Small-Mission Programmatic Strategy.

Status	Astrophysics	Space Physics	Solar System Exploration	Earth Sciences	Microgravity and Life Sciences
Existing	Explorers SMEX	Explorers SMEX		Earth Probes	Middeck payloads SMIDEX Small and rapid-response payloads
New	UNEX	Space physics smallsats	Discovery program MESUR Outer planet probes	Second-generation Earth Probes	Space Station Freedom EXPRESS

and special-topic investigations. The first of these, the Solar, Anomalous, and Magnetospheric Particle Explorer (SAMPEX), was launched this past July to study cosmic rays. This Explorer was developed within three years at a cost of \$35 million. Two other small Explorers are already in development, one for launch in 1994 and one in 1995.

The astrophysics and space physics programs were both using Delta-class explorers before this new strategy was conceived. The Explorer program was designed to target specific science objectives on limited missions. The program as a whole is cost-capped on an annual basis. It has resulted in the launch of almost 70 missions. The Extreme Ultraviolet Explorer launched this year was in development well before SSAAC began articulating its recommendations for emphasizing smaller missions.

The Earth science community also put this concept to work before this year. The Earth Probes program, also Explorer class, is a series of small and moderate-sized missions designed to address highly focused problems in Earth science. This program will provide for the collection of long-term global change data sets prior to the launch of EOS. Each free-flyer instrument has a specific purpose, providing critical measurements of specific phenomena. An example is the Total Ozone Measurement System (TOMS), which will measure total ozone concentrations.

We also created the capability for focused experimentation in the microgravity and life sciences at reduced cost and with greater frequency.

The Middeck Payloads program provides accommodations in the middeck of the Space Shuttle for investigations that require limited physical resources. This program takes advantage of the unique capabilities of the Orbiter, including the ability to support late installation or early retrieval of samples or equipment (often particularly important for life sciences and microgravity science experiments). Middeck payloads involve lower costs and shorter lead times; the level of effort is somewhat similar to sounding rocket activities in terms of scope and flight frequency, but with much longer operational periods.

The Spacelab Middeck Experiments (SMIDEX) program is a similar activity. It provides racks inside the Spacelab module that convert to locker accommodations for small experiments. This rack uses a standard interface that is similar to that used on the middeck, which means that experiments can be mounted in either the Spacelab or the middeck, depending on where space is available, without reconfiguring the equipment. SMIDEX offers the same advantages as middecks, such as late access.

Also in place before this year were interdisciplinary research opportunities that are low cost and allow quick response. For example, NASA's sounding rocket program, which I mentioned a moment ago, has logged nearly 2,500 flight missions since 1959 and is an increasingly important means of achieving space science objectives. The sounding rocket program provides about 40 flight opportunities per year to space scientists in Earth science, space physics and astronomy, and microgravity research.

But over the course of the past year, OSSA has redoubled its efforts to embark on a new strategy, and this new strategy is building on those successes. A second generation of Earth Probes is in development, as are numerous missions under the SMEX and Explorer programs. The University-Led Explorer (UNEX) program is modeled after the SMEX program but will provide additional cost savings by incorporating an enhanced role for the academic community in program design and management.

In space physics, we find another excellent example of our new strategy, the Thermosphere-Ionosphere-Mesosphere Energy and Dynamics Mission (TIMED). This mission can be accomplished on two Delta-class spacecraft. However, if the instruments could be flown on five small but common satellites, there would be a substantial increase in robustness and flexibility and a reduction in overall cost. We are evaluating the potential of using a common bus, particularly Lightsat technology, to get up and running quickly (hopefully a 1998 launch) and at less cost. I expect this strategy to be expanded to encompass many, if not all, of the other moderate space physics missions recommended by SSAAC in 1991.

There are several planetary programs in development. The Discovery program is a new initiative for low-cost missions to study the inner planets. Like the other series of Explorer-class missions, the Discovery program minimizes costs by emphasizing a focused science return achieved with the simplest possible spacecraft, science payload, and mission designs. Mars Environmental Survey (MESUR) Pathfinder is a promising candidate for the first in this series; the long-studied Near-Earth Asteroid Rendezvous mission also falls into this category.

A series of missions to the outer planets will also be possible under the new strategy. The Pluto Flyby, for example, will be the lightest weight, lowest power spacecraft possible, to achieve the shortest possible trip time. The development of the Pluto Flyby mission will be one of the first to incorporate the new-technology infusion policy, the second integral piece of OSSA's new strategy, which I will highlight in a few moments.

The MESUR Network is another good example in the planetary program. A series of 16 small landers that will study the atmosphere and

soil of Mars will be dispatched through multiple launches over a four-year period. The risk is greatly spread out — no one launch can impact the entire program, and in the case of a failure of a lander, the network will continue to be viable.

We are also looking at how to conduct high-quality investigations aboard Space Station Freedom in the quickest, most efficient manner. Two components of current planning in microgravity science are good examples. One is the development of standardized drawers or middeck lockers integrated into multiple payload racks, which will greatly reduce the complexity of integrating hardware for specific investigations. Secondly, a study is being conducted on a new type of rack called EXPRESS — Expedite the Processing of Experiments to Space Station. This effort is designed to simplify on-orbit changeout for individual experiment modules. Sub-rack payloads (which now constitute 64 percent of the total) can be integrated with minimal resource and crew training requirements by using precertified hardware and a standard integration process.

Tightly interwoven with these new and existing flight plans will be OSSA's new-technology strategy, the second component of the strategy for implementing our new vision. In the past, to conduct a mission at the least possible cost, off-the-shelf technology (that is, technology developed for other purposes) was applied to the development of new spacecraft and systems. With fewer new technologies being developed by the Department of Defense and other agencies, there is limited, if any, off-the-shelf technology to be used by the space science program. We now have to develop our own new technologies to remain state of the art and achieve our science objectives.

Testbeds are a critical component of our technology strategy for complete laboratory evaluation of spacecraft and instrument system improvement. We will reduce costs by developing generic testbeds at JPL and NASA Goddard Space Flight Center. Currently, testbeds are created on a program-specific basis — for example, testing the Cassini spacecraft. We are now going to develop testbeds that will survive individual programs, becoming permanent facilities, to allow for technological upgrades on an ongoing basis — essentially a product improvement program.

Flight opportunities for technology evaluation are also critical in the new-technology strategy. Flight opportunities will become available

aboard spacecraft and also through the use of expendable launch vehicles (ELVs). The MESUR Pathfinder mission is the ultimate spacecraft opportunity — it will serve as a technology and engineering testbed for the MESUR Network program and will hopefully include a newly developed microrover. The ELV technology infusion program is a cooperative effort between NASA, the Air Force, and industry, whereby technologies that reduce cost and increase reliability and operability are funded for development by industry for infusion within one to two years. Secondary payloads aboard ELVs, developed to meet a prelaunch integration schedule of as little as nine months, will provide other opportunities to evaluate new technologies in flight.

The result of this new-technology policy is that each mission will contribute to the advancement of spaceflight technology, to ensure that new technologies continue to become available for use on future OSSA missions. These new technologies will be new designs or techniques applied not only to spacecraft but also potentially to instruments, propulsion systems, launchers, software, and, most importantly to this audience, ground operations. The technologies must significantly improve cost, performance, or reliability and must never have flown before. The strategy is that each new technology will become available to use on future OSSA missions.

The implementation of the flight program and technology strategies that I've outlined for you this morning will result in a launch manifest that reflects a substantially increased flight rate for space science missions. It will lead to focused, well-defined science missions, many of which will close specific gaps in research remaining from the large missions of the past. It will also provide the scientific community in all the space science disciplines with predictable and frequent access to space, including deep space, in spite of the highly constrained fiscal environment. And finally, it will establish NASA's leadership in the emerging technologies of small, capable spacecraft.

One can picture a very robust launch schedule as a result of this new strategy. For example, a successful implementation would yield

- A Small Explorer launch each year.
- A Delta-class Explorer launch every other year.
- A UNEX launch each year.
- A small mission to the inner planets, under the Discovery program, every other year.

- A MESUR launch at every Mars opportunity, or approximately once every other year.
- A Pluto Fast Flyby mission followed by small outer planets missions every other year.
- An Earth Probes launch each year.
- Microgravity and life sciences missions on Spacelab and Space Station Freedom with a frequency equivalent to more than three Space Shuttle missions each year.
- A continuation of the current average of one intermediate or major mission per year.

These launches will represent a very significant level of activity each year in space science, rivaling 1992 with nine to twelve per year. And we will be able to adjust the total launch rate per year very easily as resources allow.

The overall purpose of all OSSA's efforts to date has been to free resources for maximizing the space science program in a tough fiscal environment. It should be clear that what I've said here this morning characterizes an environment in which, in addition to changing the way we design missions and develop new technology, continuous improvement in operating systems will be required. We no longer have the resources to simply multiply our operating cost by the number of spacecraft that we launch. MO&DA is coming under the same budget pressures as are the programs in development — there are growing requirements, but a flat level of resources available, which means that the MO&DA share of the total OSSA program is likely to decrease. This will mean that we must look for and implement approaches for improving efficiency across the board.

The challenge to you, therefore, is to make yourselves and your specialties part of the solution rather than part of the problem. Investments in new operating systems will continue to be difficult to justify without any demonstrable contribution to these goals. But that depends on you. No one knows better than you where improvements can best be made to ensure program viability well into the future.

I hope your efforts at this symposium are productive and rewarding. We at OSSA look forward to seeing the results.

Thank you.

PLENARY SPEECHES

COMMONALITY OF FLIGHT CONTROL SYSTEMS FOR SUPPORT OF EUROPEAN TELECOMMUNICATIONS MISSIONS

KURT DEBATIN

Head, Mission Operations Department
European Space Agency, European Space Operations Centre

This paper is concerned with the presentation of mission-independent software systems that provide a common software platform to ground data systems for mission operations. The objectives of such common software platforms are to reduce the cost of the development of mission-dedicated software systems and to increase the level of reliability of the ground data systems for mission operations.

In accordance with this objective, the Multi-Satellite Support System (MSSS) was developed at the European Space Operations Centre (ESOC). Between 1975 and 1992, the MSSS provided support to 16 European Space Agency (ESA) missions, among them very demanding science missions such as GEOS, EXOSAT, and Giotto. The successful support of these missions proved the validity of the MSSS concept with its extended mission-independent platform. This paper describes the MSSS concept and focuses on the wide use of MSSS as a flight control system for geosynchronous telecommunications satellites.

Reference is made to more than 15 telecommunications missions that are operated from Western Europe using flight control systems with an underlying MSSS concept, demonstrating the benefits of a commonly used software platform.

Finally, the paper outlines the design of the new generation of flight control systems, which is being developed at ESOC for this decade, following a period of more than 15 years of MSSS support.

BACKGROUND

At present, some 30 telecommunications satellites are being operated in Western Europe. These satellites provide TV and radio broadcasting services, telephone and data traffic, and other special services. The satellites are all positioned in

a geosynchronous orbit. The different satellites, their missions, and the organisations providing the services are listed in Table 1. The list cannot claim completeness and does not include missions that provide military services.

At the functional and technical levels, the ground segment of a telecommunication mission can be divided between the ground infrastructure, which supports the services to the user community, and the ground control system, which supports the operation and control of the orbiting spacecraft.

The ground control system basically comprises the flight control system and the telemetry, tracking, and command (TT&C) ground station. The TT&C station links the flight control system with the orbiting spacecraft. It supports telemetry acquisition, command uplink, and range measurements (see Figure 1).

For reasons of mission safety, the ground control system may be provided with backup facilities, e.g., by means of a second TT&C station and a redundant flight control system. The function of flight control systems is to provide computerised support for the navigation of the spacecraft and for monitoring and control of the spacecraft systems. Spacecraft navigation includes orbit maintenance and repositioning of the spacecraft in the geosynchronous ring.

The 30 telecommunication satellites are supported by some 14 flight control systems. Most of these systems support several missions in parallel. In such cases, one flight control system may be linked with several TT&C stations at geographically strategic positions.

Given the large commonality of flight control system requirements of the different telecommunications missions, it would seem most cost effective to base the development of the various flight control systems on a common software and hardware

Table 1. West European Telecommunications Satellites.

Satellite	Position	Mission	Service Provider	Operator, Location
MARECS-A	22.5° E	FSS	ESA/Inmarsat	ESA, Darmstadt (D)
MARECS-B2	15.0° W	FSS	ESA/Inmarsat	ESA, Darmstadt (D)
Inmarsat II-F1	64.0° E	FSS	Inmarsat	Inmarsat, London (UK)
Inmarsat II-F2	344.0° E	FSS	Inmarsat	Inmarsat, London (UK)
Inmarsat II-F3	178.0° E	FSS	Inmarsat	Inmarsat, London (UK)
Inmarsat II-F4	306.0° E	FSS	Inmarsat	Inmarsat, London (UK)
ECS-1	25.5° E	FSS	Eutelsat	ESA, Redu (B)
ECS-2	2.0° E	FSS	Eutelsat	ESA, Redu (B)
ECS-4	25.3° E	FSS	Eutelsat	ESA, Redu (B)
ECS-5	21.5° E	FSS	Eutelsat	ESA, Redu (B)
Eutelsat II-F1	13.0° E	FSS	Eutelsat	Eutelsat, Paris (F)
Eutelsat II-F2	10.0° E	FSS	Eutelsat	Eutelsat, Paris (F)
Eutelsat II-F3	16.0° E	FSS	Eutelsat	Eutelsat, Paris (F)
Eutelsat II-F4	7.0° E	FSS	Eutelsat	Eutelsat, Paris (F)
Olympus	19.0° W	DBS/FSS	ESA	ESA, Fucino (I)
Italsat	13.0° E	FSS	ASI	Telespazio, Fucino (I)
DFS Kopernikus I	23.5° E	FSS	German Telecom	German Telecom, Usingen (G)
DFS Kopernikus II	28.5° E	FSS	German Telecom	German Telecom, Usingen (G)
TV-SAT 2	19.2° W	DBS	German Telecom	German Telecom, Usingen (G)
Astra 1A	19.2° E	DBS	SES	SES, Betzdorf (L)
Astra 1B	19.2° E	DBS	SES	SES, Betzdorf (L)
Marco Polo 1	31.0° W	DBS	BSB	BSB, Southampton (UK)
Marco Polo 2	31.0° W	DBS	BSB	BSB, Southampton (UK)
Tele-X	5.0° E	DBS	Filmnet/SSC	SSC, Kiruna (S)
Telecom 2A	8.0° W	FSS	French Telecom	CNES, Toulouse (F)
Telecom 2B	5.0° W	FSS	French Telecom	CNES, Toulouse (F)
Telecom 1C	5.0° W	FSS	French Telecom	CNES, Toulouse (F)
TDF 1	19.0° W	DBS	TDF	CNES, Toulouse (F)
TDF 2	19.0° W	DBS	TDF	CNES, Toulouse (F)

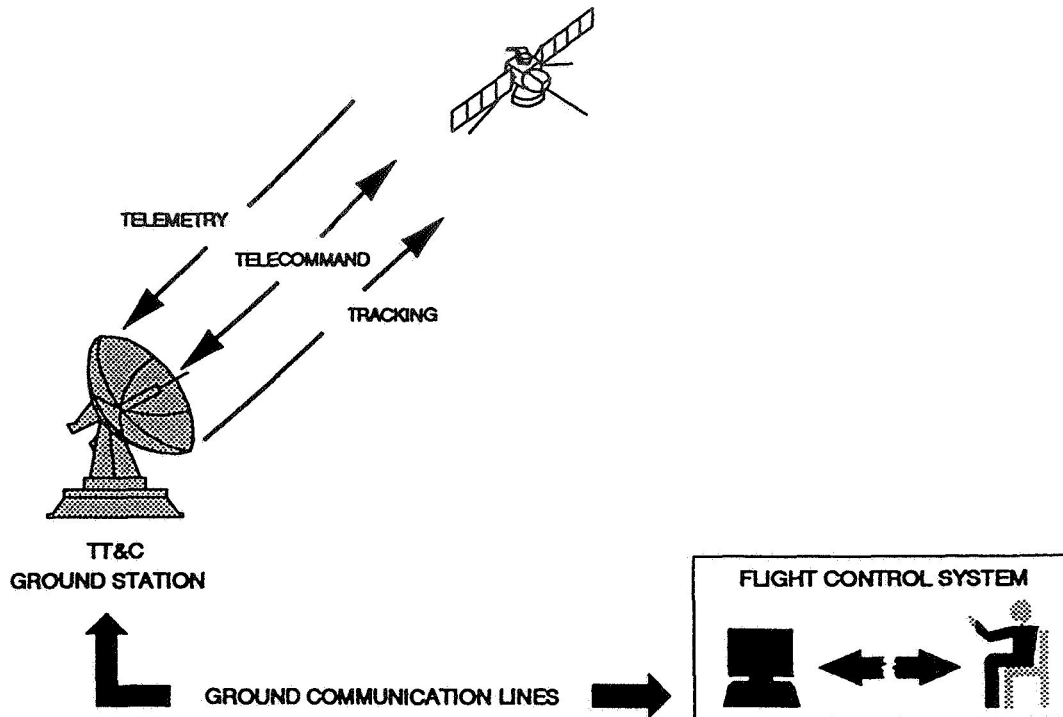


Figure 1. Ground Control System.

platform, bearing in mind that stand-alone software development of a flight control system may well exceed the equivalent effort of 50 man-years.

In reality, however, other factors very often dictate the development of a flight control system: procurement and industrial politics, in-house strategies, contractual constraints, and last, but not least, the “re-invent the wheel” syndrome.

This paper reports on a very fruitful cooperation between the European Space Agency (ESA), telecommunications space agencies, and the European software industry, materialising in a commonly used flight control system concept based on ESA’s Multi-Satellite Support System (MSSS).

MSSS DESIGN

MSSS was conceived with two prime objectives in mind:

- Provide a flight control system with the capability of supporting several satellite missions simultaneously, the number of supported missions being limited by the capacity of the underlying computer hardware platform only.
- Supply the flight control system with a high degree of mission-independent functionality, so as to minimise the need for additional, mission-specific implementations.

The overall system structure that has evolved in response to these objectives is shown in Figure 2. It consists of the following major components:

- Spacecraft database
- Functional subsystems
- MSSS configuration control database
- Support subsystems
- Archive

The functional subsystems and the support subsystems comprise generic, i.e., largely mission-independent, processing tasks. The tasks of the functional subsystems include references to the tables in the spacecraft database that describe the spacecraft and mission-specific parameters. The tasks also have access to the tables of the MSSS configuration control database that define the mission-specific processing environment upon initialisation of a task. This means that the tasks of the functional subsystems can be regarded as engines which are driven by the tables of the databases.

This table-driven approach ensures a high degree of flexibility. Additional support of a new satellite mission does not necessitate cumbersome

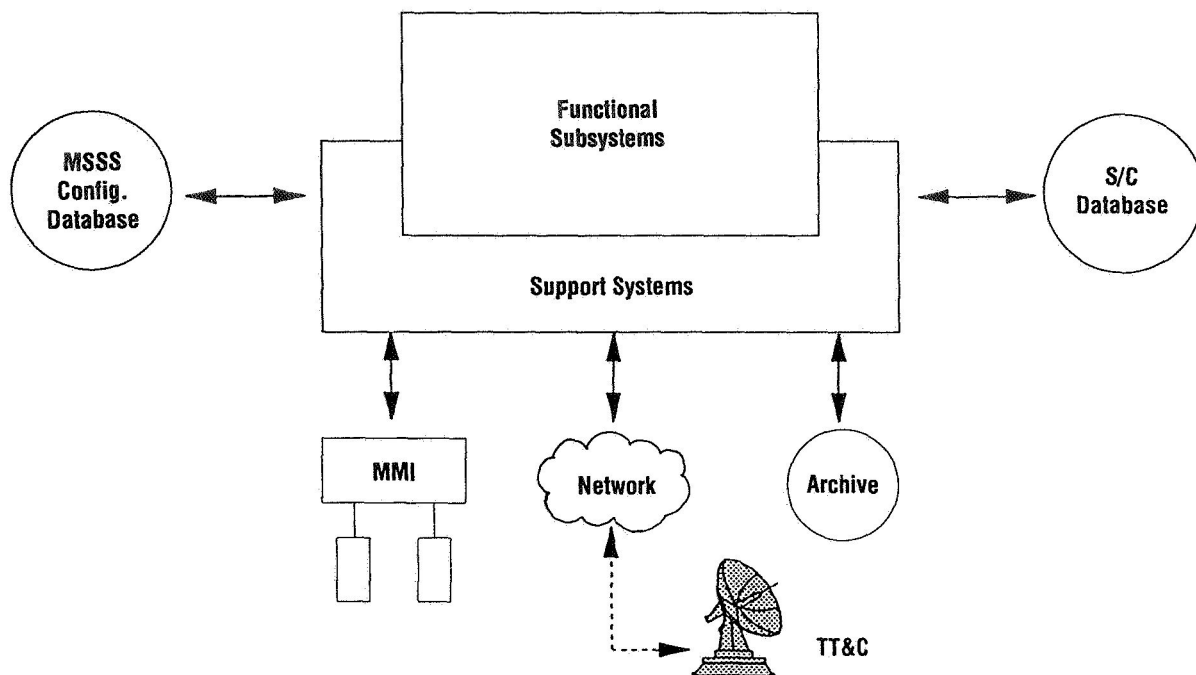
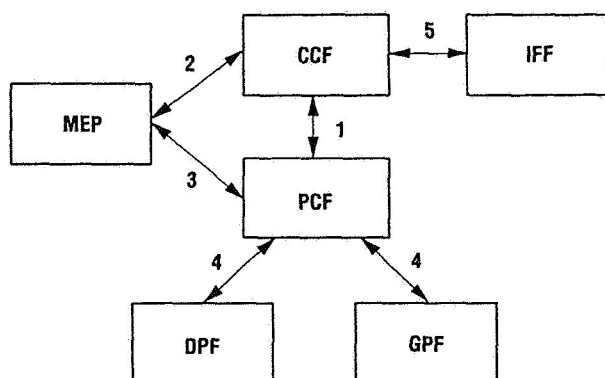


Figure 2. Functional Diagram of MSSS.



Relationship Between Files

- | | |
|---|---|
| 1. Command verification against telemetry parameters. | status checks of telemetry parameters dependent upon mode of operation. |
| 2. Command validation dependent upon mode of operation. | 4. Telemetry parameters included in display pro formae. |
| 3. Telemetry parameters appear as operands in mode equations; limit and | 5. Extended description of command functions. |

Figure 3. Files of the Spacecraft Database.

and risky software changes at the functional subsystem level but only requires the incorporation of the new mission-specific parameter tables in the databases. The different components of the MSSS are described below in more detail.

THE SPACECRAFT DATABASE

For each spacecraft to be supported, the spacecraft database contains a set of files that define the characteristics of the formats, available commands, and structure of telemetry in terms of subframes and frames (see Figure 3). The files of the spacecraft database can be directly edited by the user.

The parameter characteristic file (PCF) holds the parameter details associated with the telemetry data stream from a satellite. The file identifies each parameter and provides parameter-specific information such as calibration curve factors, number of occurrences per frame and subframe, status and limit checks, and the related mode equation number.

The mode equation file (MEF) defines all mode equations applicable to a satellite mission. The operands of the mode equations refer to parameters in the PCF.

The command characteristic file (CCF) holds an entry for each spacecraft command and related information. The information describes the characteristics of the command, e.g., it specifies command validation and verification with reference to the applicable mode equation and to the telemetry parameters as defined in the MEF and PCF, respectively.

The individual function file (IFF) complements the CCF with more detailed information on specific command functions, e.g., in conjunction with memory load commands.

The display files DPF and GPF hold details of the alphanumeric and the graphics display pro formae covering also the included PCF parameters.

FUNCTIONAL SUBSYSTEMS

For each satellite mission, the functional subsystems perform the following major tasks:

- Telemetry processing, which covers the acquisition of the telemetry, processing (mode equations, derived parameters), validation (out of limits), and filing.
- Telecommanding, which includes the transmission of telecommands and their validation and verification; and the manual, scheduled, and automatic generation of commands or sequences of commands.
- Information display, which covers all types of displays of real-time data and retrieved data in accordance with the pro formae as described in the display files of the spacecraft database.
- Archiving of telemetry and command history data.
- Acquisition of ranging data and transmission of antenna steering information to ground stations.
- Remote station monitoring and control.

CONFIGURATION CONTROL DATABASE

The MSSS configuration control database contains a set of tables that define the processing environment for the tasks of the functional subsystems. For example, the environment may define the routing and the layout of a spacecraft data stream by format and size of subframes and frames, and by the included parameter identities. At initialisation, the tasks read the relevant tables.

In addition to this relatively static definition of task-oriented data processing environments, the

database also includes a file that dynamically registers parameter processing and data routing options and associated changes so that they can take effect in near real time, i.e., during the execution of a task.

SUPPORT SUBSYSTEMS

The support subsystems comprise tasks that provide common services to the functional subsystems. These services include buffer management, record filing and access routines to databases, management of dynamic configuration tables in memory, error detection and recording, event scheduling, etc.

In a wider sense, the subsystems also provide middleware functions such as console drivers, display form filling, and direct access methods.

ARCHIVE

The archive holds the telemetry and command history files.

USE OF MSSS AS A FLIGHT CONTROL SYSTEM

The validity of the MSSS concept with its parameter table structure is in particular demonstrated by its wide use as a flight control system in support of telecommunications missions (see Figure 4). Here it has set a de facto European standard.

The first telecommunications mission supported by MSSS was OTS, ESA's first telecommunication test satellite. MSSS was then also used for operational support of two maritime missions, MARECS-A and MARECS-B2. The two missions are still operated on behalf of Inmarsat from ESA's European Space Operations Centre (ESOC) in Darmstadt, Germany.

Another series of telecommunications missions, ECS-1, ECS-2, ECS-4, and ECS-5, are also being supported by MSSS. The four missions are operated for Eutelsat from ESA's Control Centre at Redu, Belgium.

Initially, the MSSS hardware platform was based on two back-end computers from CII and front-end processors from Siemens. Between 1981 and 1986, the platform was gradually changed. The CII computers were replaced by more powerful computers of type Encore, and then the Siemens processors were removed and new Intel-based workstations were added. These changes have

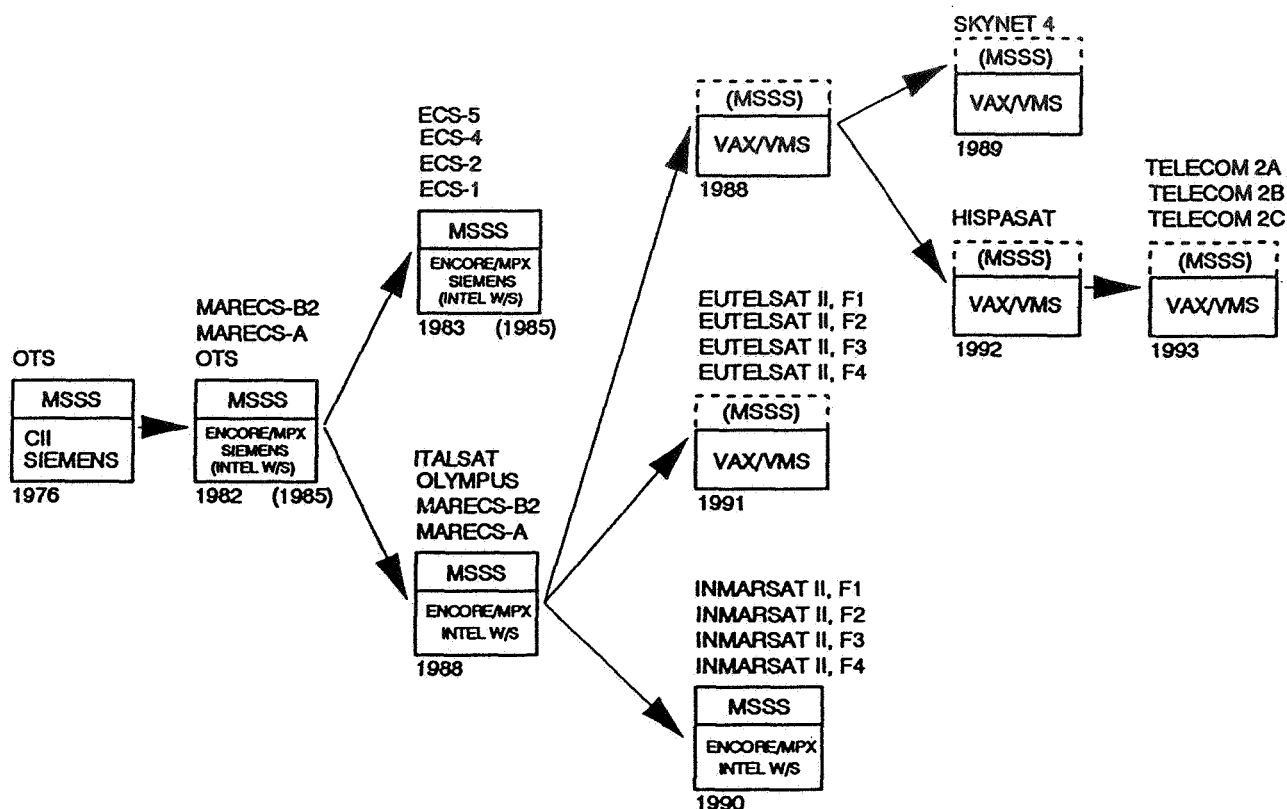


Figure 4. MSSS History.

resulted in a substantial increase of the MSSS performance, providing sufficient capacity to cope with the parallel support of at least six satellite missions of the ECS class.

The next telecommunications mission supported by MSSS was Olympus-F1, ESA's second telecommunications test satellite. To support this highly complex satellite weighing 1.4 tons, the functionality of MSSS had to be upgraded. The upgrades encompassed considerable extensions to the command and telemetry table structure in order to accommodate additional block commands and an extended set of telemetry parameter references. The system also had to be provided with a facility for onboard software maintenance and configuration control. Furthermore, the onboard generation of parameters in dwell mode called for the implementation of special telemetry data processing and display modules.

The Olympus upgrades required a software development effort equivalent to some 20 man-years, which, even though representing a significant expense, was considerably less than the cost

of development from scratch. On the other hand, these investments broadened the functionality and the maturity of MSSS to such an extent that it was able to meet the requirements of other, more advanced telecommunications missions. As a consequence, only minor adaptations had to be made to MSSS for the support of the Italian space agency's Italsat. This satellite, which provides a capacity of approximately 12,000 telephone circuits, was operated during LEOP and during several months following LEOP from ESOC before control was transferred to the Italian Control Centre in Fucino.

In 1989, Inmarsat decided to use MSSS as the flight control system for their new series of satellite missions: Inmarsat II-F1, II-F2, II-F3, and II-F4. The installation and commissioning of MSSS at the London Inmarsat Control Centre was constrained by a rather tight time schedule. It included as major activities the procurement and installation of the computer hardware, the implementation of the MSSS software, and the parametric configuration

of the table structure in accordance with the specific mission requirements. In addition, a number of modifications and enhancements to the system had to be implemented. They were mainly concerned with specific spacecraft requirements, e.g., onboard, time-tagged command control and command confirmation; support of telecommand blocks; and dwell telemetry parameter displays. Also, station monitoring and control facilities were implemented with the same functionality as was provided for spacecraft monitoring and control. After one year, all tasks, including the enhancements, were completed and MSSS was ready for operation. The system is currently supporting the four Inmarsat missions and four TT&C stations.

European software companies have also adopted the MSSS concept for the further development of flight control systems. In this way, their significant involvement in the development of MSSS at ESOC has come to fruition.

Science Systems Ltd., which received an MSSS software licence from ESA, transformed MSSS to a new platform on VAX/VMS computers from DEC. The new system, known as the Kernel TT&C System, was used as the basis of a flight control system which Science Systems Ltd. developed for the British Ministry of Defence for the support of Skynet 4 and NATO IV. It was implemented as a distributed system on VAX machines and was provided with extensive redundancy and flexible reconfiguration capabilities at the basic system level to ensure the required high degree of system availability. The workstations were provided with advanced display facilities based on X-Windows and GKS graphics software. This system in support of Skynet has been in operation since 1990 and provides services to the military.

Another implementation of the Kernel system was specifically tailored to the needs of the Eurostar platform. This involved, in particular, considerable enhancements to the telecommand subsystem. This new version provides the basis of the flight control system that supports the Hispasat mission from the Arganda Control Centre near Madrid, Spain. It will also constitute the core of the flight control system which will be used for the support of the French Telecom 2A, 2B, and 2C missions.

In parallel to this, the British firm Logica, based on an MSSS licence from ESA and in response to an order from Eutelsat, has also per-

formed an MSSS-based implementation on a hardware platform from DEC consisting of a VAX range of computers and VAX workstations. The MSSS table structure was implemented on a relational database from Ingres. The major changes in the implementation by Logica are concerned with the provision of sophisticated display facilities on the workstations and with the processing and display of the spacecraft data stream at the level of frames rather than subframes. An only slightly modified version of this implementation will be used as a flight control system for the support of Turksat.

In conclusion, up to 20 telecommunications missions have been supported by flight control systems with an MSSS-based architecture. It has to be recognised, however, that the diverse systems show considerable variations. Incremental system changes of a more or less substantial nature have been introduced since 1976, when MSSS was used for the first time. This is not surprising, when one considers the different technical, contractual, and political frame conditions under which these systems have been implemented during a period of more than 15 years.

Different operations concepts have evolved, which have in particular brought about considerable extensions of the display facilities on the workstations. Major system changes arose from different spacecraft designs and mission requirements in combination with the changeover of the computer platform from Encore/MPS to VAX/VMS. Notwithstanding these differences, the various systems can be traced back to a common architecture, and this has reduced cost and technical risks far below the level that normally has to be assumed for new and independent developments.

Associated with MSSS, the Portable ESOC Package for Synchronous Orbit Control (PEPSOC) was developed at ESOC. It is used for orbital control of geosynchronous satellite missions, i.e., it determines the orbital position of the spacecraft by means of range measurements and specifies the orbit manoeuvres (velocity increment vectors) for repositioning the spacecraft.

PEPSOC is a software package that runs on a workstation or on a PC and can be easily interfaced with any flight control system. PEPSOC is

recognised as a standard tool by the European space community, not only in connection with MSSS-based systems, as proven by more than 17 PEPSOC licences that have been granted to industry and space agencies.

FUTURE OUTLOOK

The future evolution of flight control systems will be determined by the increasing complexity of spacecraft systems and by the introduction of innovative informatics technologies.

ESOC has given due consideration to this foreseeable evolution and has started with the development of the architecture for a new generation of flight control systems. Following the MSSS success story, the new generation, named Spacecraft Operations System (SCOS), is intended to cover the agency's requirements over the next decade for the support of different classes of missions, including telecommunications missions and highly complex science missions.

In essence, the new architecture features the following elements:

- Packetised telemetry and telecommanding according to the defined CCSDS standards.
- Object-oriented approach.
- UNIX-based platform.

The handling of telemetry and command packets in combination with the object-oriented processing approach will provide the user with more efficient and comprehensive access to the functionality of the individual subsystems and units of the orbiting spacecraft. The database will

hold operational procedures and associated data that can be addressed as concise entities. In many respects, the changeover from MSSS to SCOS can be compared with the parallel technological move from a relational database to an object-oriented database approach.

Similar to MSSS, SCOS will provide a large, mission-independent system platform. This will be accomplished by taking advantage of the inheritance feature of object-oriented modules and by a possible later introduction of packet utilisation standards, which will define the data files in the TM/TC packets.

The choice of UNIX as the operating system is obvious because of its worldwide recognition as an international standard. Although UNIX is not conceived as a real-time operating system, there now exist UNIX versions that offer real-time capabilities. The UNIX platform will be structured around a network of UNIX-supported workstations. The integration of the workstations into a LAN-based structure with distributed client and server functions provides an environment that can be easily configured to the needs of different classes of missions and in the short term to different mission scenarios.

The workstations provide the capacity for the support of the required processing and database functions, and advanced human-machine interfaces based on the UNIX operating system. UNIX will ensure portability of SCOS between different hardware platforms and thus removes a traditional problem associated with MSSS.

NASDA'S VIEW OF GROUND CONTROL IN MISSION OPERATIONS

N 9 4 - 2 3 8 3 5

SATOSHI TATENO

Executive Director

National Space Development Agency of Japan

This paper presents an overview of the present status and future plans of the National Space Development Agency of Japan's (NASDA's) ground segment and related space missions. The described ground segment consists of the tracking and data acquisition (T&DA) system and the Earth Observation Center (EOC) system. In addition to these systems, the current plan of the Engineering Support Center (ESC) for the Japanese Experiment Module (JEM) attached to Space Station Freedom is introduced. Then, NASDA's fundamental point of view on the future trend of operations and technologies in the coming new space era is discussed. Within the discussion, the increasing importance of international cooperation is also mentioned.

INTRODUCTION

The National Space Development Agency of Japan (NASDA), established in 1969, is one of the two major space organizations in Japan; the other is the Institute of Space and Astronautical Science (ISAS). NASDA is responsible for developing the space system for practical use — meteorological observations, communications, broadcasting, and observation of Earth's environment.

On the other hand, the objective of ISAS is to do research in space science using satellites and rockets. It has a long history (since 1964) and has achieved excellent results in space science. ISAS has its own tracking and data acquisition (T&DA) capabilities, but NASDA provides orbit-determination results for low-Earth-orbit satellites.

In accordance with NASDA's charter, initially the agency devoted all its efforts toward realizing geosynchronous missions for practical use and has placed many satellites into orbit since 1977. After several experimental mission successes, the Japanese people have received full service from geosynchronous missions.

The next challenge for NASDA was to develop the Earth observation satellite series, and MOS-1 and MOS-1b were successfully placed into Sun-synchronous orbits in 1987 and 1990, respectively. The Japanese Earth Resources Satellite (JERS-1), carrying synthetic-aperture radar (SAR),

was successfully put into orbit in 1992. Those satellites' data have been available to foreign space agencies since the MOS-1 launch. In parallel with those missions, NASDA has launched five Engineering Test Satellites (ETSs) since 1975 in order to verify new space technologies.

The corresponding ground support system has also been gradually developed since the early 1970s. NASDA's present ground segments for space missions consist of a spacecraft T&DA system and an Earth Observation Center (EOC) system. The T&DA system consists of the Tracking and Control Center (TACC), established in 1975, which is the focal point of satellite control and related operations, and four Tracking and Control Stations (TACs), including the station at Kiruna, Sweden.

The EOC was initially established in 1978 as a center of Earth observation in Japan to receive, process, and distribute Landsat data. Since then, its capability has been enhanced to cope with both Japanese and foreign Earth observation satellite evolution.

Another new aspect of development in the ground segment is the Engineering Support Center (ESC) system of the Japanese Experiment Module (JEM) to be attached to Space Station Freedom (SSF), in which Japan is participating with international partners — the National Aeronautics and Space Administration (NASA), the European Space Agency (ESA), and Canada.

As typically presented in the JEM project, the principal objectives of NASDA's space development have been dramatically changing since the late 1980s. The objectives have become to develop space as a new frontier of human activity with international partners — the "Mission from Planet Earth" missions.

Other types of missions are to utilize the space environment to produce new materials, to make new experiments useful for human life, and to monitor Earth's environment and resources globally from satellites in orbit — "Mission to Planet Earth" projects. In order to support these

new types of space projects, the present ground segment also has to be changed and its capability enhanced in accordance with space segment developments.

In the following sections, NASDA's present and future spacecraft missions, related systems of the ground segment, and required technical research and development are briefly introduced.

SPACE MISSIONS OVERVIEW

In this section, the present NASDA spacecraft launch schedule, in which some missions are still under investigation, and the typical missions which will have great impact on the ground segments are introduced (see Figure 1).

THE ETSS

The first ETS was launched in 1975; NASDA has placed five such satellites into orbit to verify new space technologies. The fifth satellite — ETS-V — is still operational (since 1987) and is scheduled for a tele-education experiment within Asian countries as an International Space Year (ISY) campaign. At present, NASDA has three satellite projects that have been funded. Each mission is introduced briefly in the following sections.

ETS-VI

This is NASDA's largest geosynchronous satellite so far and will be launched in mid-1994 by the H-II launch vehicle. It has two major goals — to verify basic technologies of bus equipment for large-scale satellites and to verify wide-range communication technology from space network to mobile communication, as follows:

- Fixed and mobile communication
- Intersatellite communication: K-band, S-band, and feeder link
- O-band communication
- Laser communication

From the ground segment point of view, ETS-VI brings two types of new technologies: the space network and the end-user-oriented Operations Control Center (OCC) capability. Figure 2 shows the ETS-VI.

COMETS

COMETS is scheduled for launch in 1997. The major missions are listed below, except for bus technologies.

- Intersatellite communication
- 22-GHz broadcasting
- Ka-band mobile communication

COMETS will enhance NASDA's Space Network from ETS-VI and the system is expected to support such satellites as the Advanced Earth Observation Satellite (ADEOS) and ETS-VII.

Calendar Year	9 2	9 3	9 4	9 5	9 6	9 7	9 8	9 9	2 0 0 0
Launch Vehicle • H-II • J-I			▲ TF-1 ▲ TF-2	▲ TF-3	△ TF-1				△ HOPE: H-II Orbiting Plane
ETS Series, etc			▲ VEP ▲ ETS-VI			▲ COMETS ▲ ETS-VII	▲ OICETS		△ DRTSS
Earth Environment Monitor	▲ JERS-1			▲ GMS-5	▲ ADEOS				△ ADEOS-II
JEM/SSF					▲ JFD		▲ JEM#1	▲ JEM#2	
Other Missions			▲ OREX (Re-entry Exp)	▲ SFU/ISAS △ TBD (Re-entry Exp)		▲ TRAM/NASA			▲ Scheduled launch △ Under study

Figure 1. NASDA's Spacecraft Launch Schedule.

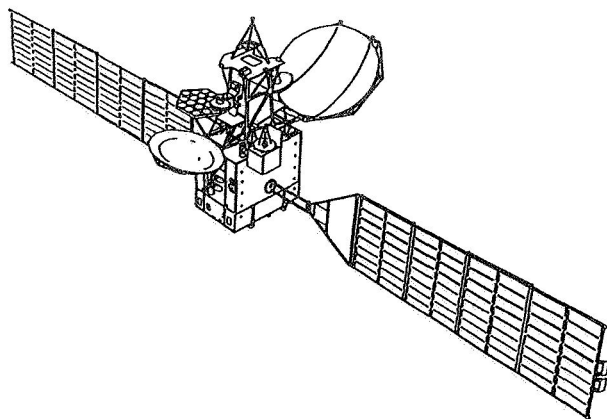


Figure 2. ETS-VI.

ETS-VII

The ETS-VII (see Figure 3) is the most challenging spacecraft ever planned by NASDA. It has two major missions: the unmanned Rendezvous and Docking Experiment and the Remote Controlled Space Robotics Experiment.

ETS-VII is composed of two satellites — Chaser and Target. The system is cooperatively controlled by the onboard computer and the ground system. The required technologies to be developed and verified through this mission will be a foundation for realizing a space transportation system in the early 21st century, except for the re-entry technology. The required ground segments also have many new challenging developments. For instance, NASDA needs to develop the technologies such as teleoperation, simultaneous two-satellite cooperative control, control via space network, and so on.

OTHER PLANS

The ETS-type satellites are very important for proving new technologies in space with low cost. So, NASDA is developing a new, relatively small launcher named J-1, which will be in service by 1996. As a first candidate small satellite, an experimental satellite for optical intersatellite communications is under study.

EARTH OBSERVATION PROJECTS

An Earth observation satellite has had a very important place in NASDA since the first launch of MOS-1 in 1987, by contributing to the observation of Earth's global environment and natural resources from space. The projects to follow JERS-1, launched this year, are introduced below.

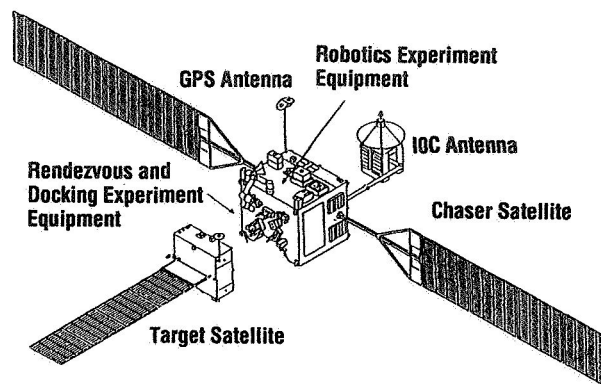


Figure 3. ETS-VII.

ADEOS

ADEOS is a large-scale, composite-type satellite that will observe Earth's environment globally with several instruments. Two instruments — TOMS and NSCAT — are to be delivered by NASA, and POLDER will be delivered by the French space agency, Centre National d'Études Spatiales (CNES).

From an operations point of view, ADEOS will be the first Japanese satellite that can transfer data via the data-relay capability of COMETS. In order to efficiently operate payloads and to deliver observed data quickly to the end users, the OCC system has to have complicated scheduling functions and the EOC system needs to have the capability to quickly process and deliver a large quantity of data by the launch date.

FOLLOW-ON PROJECTS

The follow-on project to ADEOS, ADEOS-II, is under study and, after authorization by the Japanese government, the announcement of opportunity will be published to investigators both inside and outside of Japan. Like its predecessor, ADEOS-II will also be an international cooperative mission and will carry foreign users' equipment.

By the early 21st century, NASDA's Earth observing satellites will form a system with other agencies' satellites to supplement each other's capabilities through international cooperation.

SPACE STATION

In the SSF project, Japan will provide the JEM, which will be attached to the Space Station. The JEM consists of four major parts — pressurized module, experiment logistic module, exposed

facilities, and manipulator (see Figure 4). JEM is scheduled to be put into orbit by NASA's Space Shuttle around 1998. In order to support JEM and the Japanese payload operations through experiments, and in case of anomaly, NASDA plans to develop an ESC in Japan.

GROUND SEGMENT

Based on the requirements from the present and future missions described above, NASDA's ground segment is continuously evolving and being enhanced. The status and future plans of NASDA's ground segment and related development and studies are introduced below.

T&DA SYSTEM

The T&DA system consists of a TACC and four TACSs; one of the TACSs is at Esrange, part of the Swedish Space Corporation. The TACC is the focal point of NASDA's satellite control and has the principal roles, namely, operations control of satellites, T&DA network control, and supporting computation, including flight dynamics. Major satellites in operation are the Earth observation missions (MOS-1/1b and JERS-1) and ETS-V. The OCC of Earth observation at TACC keeps in close contact with the EOC in order to achieve full capabilities according to the requirements from mission users. An overview of related systems and plans are discussed in the following sections.

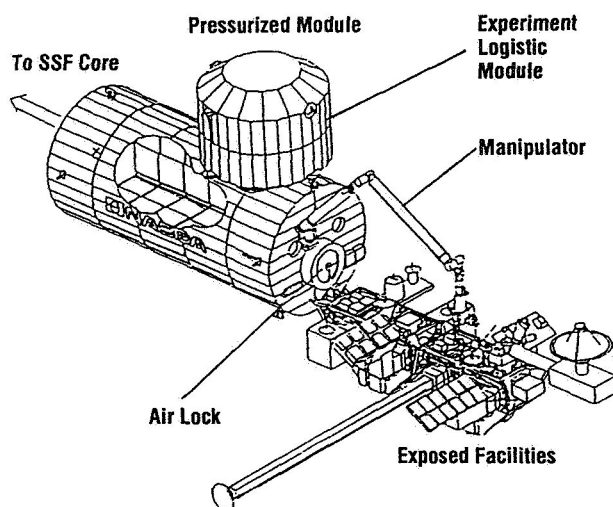


Figure 4. JEM.

GROUND NETWORK

The ground network consists of four TACSs; each TACS has two sets of unified S-band equipment. The network has been continuously upgraded since 1989, and a new capability to control TACS major equipment remotely from TACC is being developed. Most of the system has already been completed and the JERS-1 has been operated by the system. Implementation will be completed by the end of next April, and almost every TACS operation can be remotely performed from the TACC. This network can be connected with a foreign agency's network which is compatible with NASCOM via a gateway system. NASDA's ground network will continue to exist in the 21st century as the complementary part of the space network.

SPACE NETWORK

NASDA's space network will play a very important role in future space operations. The first space network experiment is to be started in 1994 by using ETS-VI capability. The ground portion of the experimental space network system is under development at TACC in parallel with the satellite development. As the outline of the experiment in Figure 5 shows, NASDA also plans to verify interoperability of the space network with NASA's Tracking and Data Relay Satellite System (TDRSS). Then the COMETS experiment will follow in 1997. As a semi-operational system, the system will support several NASDA missions in the late 1990s.

By the early 21st century, NASDA will have a fully operational space network by launching two Data Relay and Tracking Satellites (DRTSs), which will have Ka-band and S-band intersatellite communication capabilities. In order to realize an interoperable space network in the near future, NASDA works closely with both NASA and ESA and has been participating in the Space Network Interoperability Panel (SNIP) with them. SNIP coordinates radio frequencies, modulation parameters, architecture of operation, etc.

In the coming 21st century, the NASDA system will take an important place in the global space network system in support of international space missions.

OCC SYSTEM

NASDA's OCC system has been dramatically changing from the level of ETS-VI OCC system development. The OCC will have the capability to relay permitted commands and telemetry from/to several specified users' operations centers

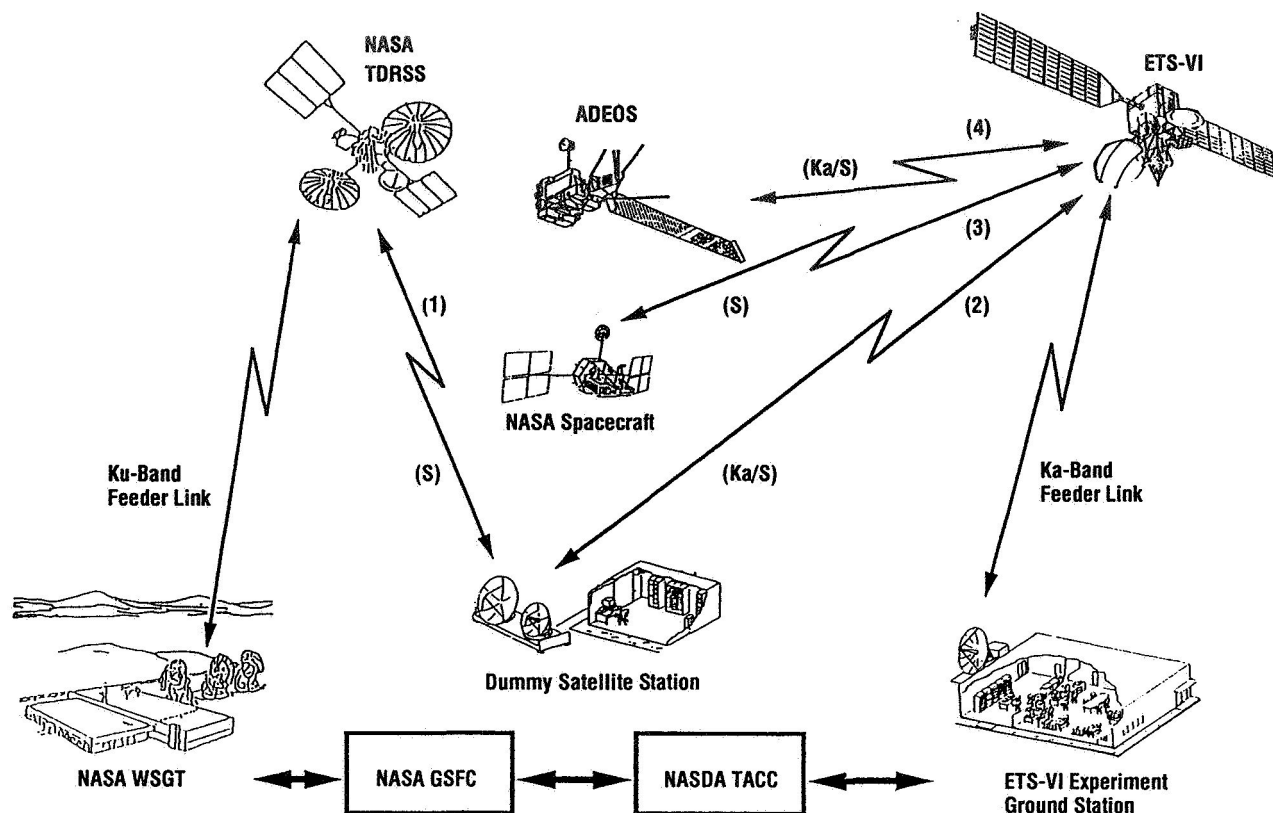


Figure 5. Space Network Experiment.

located in a remote area via the communication line. This will be the first step for NASDA to implement an end-user-oriented system. This capability will be further extended for the ETS-VII OCC system, which has to support unmanned rendezvous and docking activities and robotics experiments.

With missions becoming more complicated and capabilities of satellites growing, onboard computers will be employed to directly control some portion of the payload. This is the case of ETS-VII, and ADEOS will also carry many central processing units (CPUs) on board. This trend will bring some changes in OCC capabilities and may reduce some portion of the OCC load. However, OCC will need to cope with the complexity of future satellites. Its capabilities must be further enhanced to make detailed mission schedules in order to bring out the full capability of satellites safely, to assist ground controllers by providing

some kind of satellite simulators in case of emergency, onboard software rewriting, and so on.

Technologies to be obtained through the development of ETS-VI, ETS-VII, and ADEOS ground control systems will be the foundation of OCC technologies for more advanced 21st century spacecraft.

FLIGHT DYNAMICS SYSTEM

The flight dynamics system supports flight projects and network operations by orbit and attitude determination and control. The present concern is to develop a new system to support the space network experiment carried out by using ETS-VI and to prepare for future Moon or planetary missions.

EOC SYSTEM

The EOC has two Earth observation systems that are capable of receiving and delivering data and will have another new system by ADEOS launch in order to enhance the capability to cope with ADEOS and future missions. As a next step, a

direct data-receiving capability from the Japanese DRTS will be implemented by the time of the ADEOS-II launch in the late 1990s. At present, EOC receives data from the Earth observation satellites MOS-1/1b, JERS-1, Landsat, SPOT, and EERS-1.

NASDA is also promoting EOC as one of the Earth observation data centers of the world as a part of the activities of the Committee of Earth Observation Satellites (CEOS). The CEOS-International Directory Network (CEOS-IDN) has already been implemented and is connected to users both inside and outside of Japan and to systems belonging to foreign agencies such as NASA and ESA.

As introduced here, the EOC is becoming one of the Earth environment monitoring and data centers of the world. Present EOC capabilities are described briefly below.

MISSION CONTROL

Mission Control manages the payload operation plan based upon user request, keeping close contact with TACC. In the case of a foreign satellite, EOC summarizes Japanese users' requests and sends them to the satellite control centers.

DATA RECEIVING AND RECORDING

Data from each satellite are received by one of two antennas and recorded on high-density digital tape (HDDT) based upon the operation plan managed by Mission Control.

DATA PROCESSING AND EVALUATION

After data distortion correction by computer, data are recorded on computer-compatible tape (CCT). Image data are produced by printing on films using laser optics equipment, etc.

DATA ARCHIVING AND DELIVERING

The recorded HDDT, CCT, originally printed films, and catalogue of this information are archived and delivered according to the user's request. This capability will be greatly enhanced according to the above mentioned development of a new data center.

JEM ESC SYSTEM

NASDA plans to build the JEM ESC at the Tsukuba Space Center, which will be the first control center in Japan for manned flight and will support experiments performed at the JEM attached to the Space Station. The system will be connected to both SSOC and POIC in the United States via the international communication line.

The capabilities of the ESC are to make an initial operations plan, to assist Japanese staff in coordinating experiment plans among the partners, and to monitor and evaluate JEM functions.

Before fully implementing the ESC system, NASDA is going to verify its capability by taking advantage of the JEM Flight Demonstration (JFD), which is scheduled to verify the JEM manipulator by using NASA's Space Shuttle in 1996. The details of ESC's capability are now under study, which is being done in coordination with international partners.

OTHER CONSIDERATIONS

NASDA's ground segment will be developed gradually in order to cope with future advanced missions planning to have interoperability among space agencies. Several key points common to the realization of such a ground segment for future missions are summarized below:

- Establishment of a space communications standard.
- Establishment of an interoperable space tracking and data network.
- Establishment of a global data network that is fully open to the user community and that enables users to utilize space data and to access the payload itself safely.
- Development of a secure and manageable operations control system for both autonomous and complex spacecraft.

CONCLUSIONS

As human activity in space moves from near Earth to the Moon or Mars and as Earth's environment becomes more important to the human race in the early 21st century, more space projects will be performed through international cooperation and require cost effectiveness. As the ground segment is developed accordingly, NASDA will continue to develop related ground systems and technologies, keeping close relations with its international partners.

FUTURE TRENDS IN NASA SPACE OPERATIONS

N94-23836

CHARLES F. FUECHSEL

Director, Communications and Data Systems Division
Office of Space Communications
National Aeronautics and Space Administration

It is a great privilege for me to address my colleagues at SPACEOps 92. Today, I would like to share with you some current directions we are pursuing in NASA's Office of Space Communications (OSC).

The space agencies of the world have produced some remarkable achievements in conducting our missions. It is also true that we consume significant resources in controlling and operating our spacecraft. In these challenging economic times, the degree to which our agencies can develop new missions will depend on our ability to reduce operating costs. I believe that there are many opportunities for doing this, and I will explore some of our plans in this regard. First, I will review the business of "space operations."

WHAT ARE SPACE OPERATIONS?

We use spacecraft as our agents to collect information on our behalf. This implies the ability to communicate and to interact with our agents using some form of radio transmission. The ability to interact with spacecraft implies that a piece of the total system resides on Earth. The functions and complexity of the Earth-piece determine its cost.

For example, Hubble Space Telescope and Voyager are quite different missions, yet they are similar in operational complexity. Each requires about 10,000 commands to define a day's worth of activities. This process is labor intensive for both missions.

We can generalize — independently of specific systems — the operations functions needed by every mission in a reference model for OSC services, shown in Figure 1. The model is similar to the OSI reference model. It is hierarchical with well-defined interfaces that isolate the internal details from one layer to the next.

The components of the OSC reference model are as follows:

- Radio-link service – TDRSS, DSN, GSTDN.
- Digital bit-level functions – Bit synchronization and convolutional decoding.

- Frame-level functions – Recognizing and removing the transport structure.
- Packet functions – Recognizing and separating data substructures.
- Data processing
 - Removing artifacts of communication to recover original instrument data; error detection and correction; time reordering and overlap removal.
 - Generation of user data sets.
 - Correlation of data with spacecraft position and attitude.
- Mission control (easy to say, harder to do)
 - Assessing spacecraft performance.
 - Detecting and responding to problems.
 - Planning mission activities and scheduling radio-link services.
 - Creating the operations program and certifying its safety.
- Tracking measurement – Measuring spacecraft position (distance, angle, velocity) as a function of time.
- Tracking data processing – Removing artifacts; conversion to engineering units.
- Orbit determination – Estimating position and velocity as a function of time relative to a model and a coordinate system.
- Attitude determination – Estimating the orientation and rate of change of the spacecraft as a function of time.

Moving up the stacks of the reference model, we see that the systems reflect increasing knowledge of how individual spacecraft are built and how they work. The lower level services readily accommodate new customers with little preparation, if they meet the interfaces and capacity exists. Higher level services require the development of systems based on individual spacecraft.

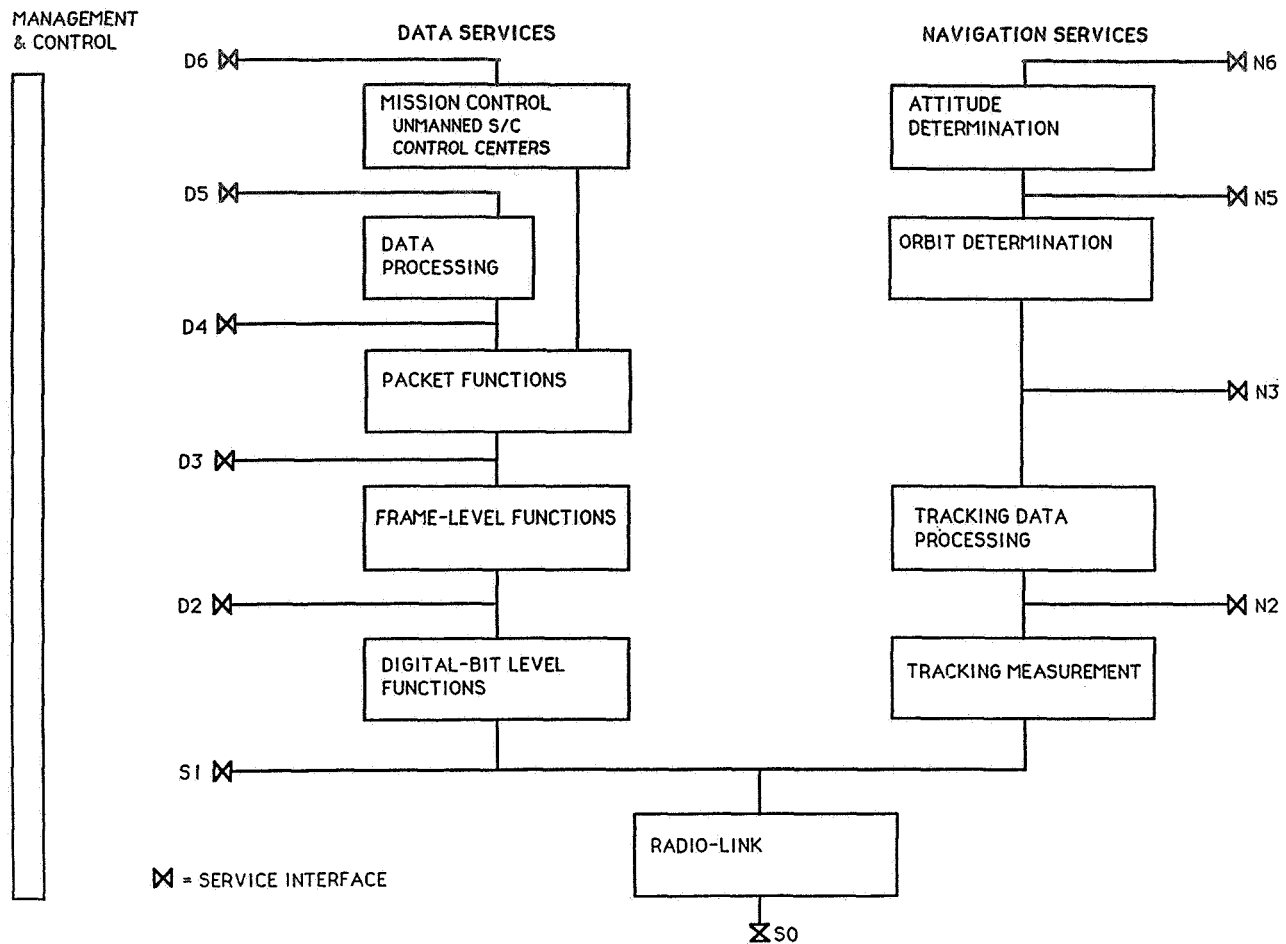


Figure 1. Reference Model for OSC Services.

ECONOMICS OF SPACE OPERATIONS

Our challenge is to provide operations services more economically — every space agency is currently faced with this. The cost of operations directly competes with the ability to develop new missions.

One way to achieve economy is to provide common solutions for recurring mission needs: for example, the Deep Space Network and the Space Network. This has been recognized for a long time and many space agencies are organized with this in mind. OSC within NASA is an example. It does not in itself minimize total cost — for example, the cost of spacecraft can be reduced by assigning more functions to the ground system. Minimal cost is achieved only if spacecraft designers are accountable for the total cost of their missions.

In order to reduce costs, we need to know where our costs are incurred.

- Lower level services: The incremental cost for a new mission is negligible, unless new capacity is

required. In this case, the cost is known and can be associated with the requirement.

- Higher level services (mission control, data processing, and flight dynamics): The incremental cost for a new mission is significant, because ground systems must reflect the design and functions of the spacecraft. There is a system development cost and an operations staff cost associated with each mission. System capacity is frequently an issue.

There is the significant possibility that spacecraft design decisions will adversely affect development and startup costs.

Our experience indicates that, for unmanned Earth-orbiters, on average:

- Development and startup costs are about twice the yearly operating cost.
- Development is quickly overtaken by operations and the latter is the biggest payoff.

OPPORTUNITIES FOR COST SAVINGS

LOWER LEVEL SERVICES

- Automation: same services operated by fewer people
 - STGT for spacecraft monitor and control and status reporting automation; reduce operations staff required for current capacity by one-third.
 - Ground networks: a new generation of low-cost transportable stations with semi-automated operations.
- New technology: modern equipment, commercial products
 - Deep Space Network: increase data rates by a factor of ten and simultaneous capacity from three to five links.

HIGHER LEVEL SERVICES

- Major thrust to standardize the services and their interfaces, so that customers can design their spacecraft accordingly. This will have a significant effect on development costs, and to some extent on operations. Currently, we frequently need to tailor systems to “as-built” spacecraft.
- Data processing
 - Flight systems now under development use standards recommended by the Consultative Committee for Space Data Systems (CCSDS).
 - Significant economies on the flight and ground sides of the data system are resulting. This is enabling the development of a new ground data processing system that will service a large number of different missions. The development activity for a new mission will consist of creating a control table to identify the virtual channels and specify processing options.
- Mission control
 - With standard services and interfaces, the development cost will be sharply lowered through reuse of systems and software.
 - This implies building our spacecraft to a similar concept of operations, representing stored command programs; even building our spacecraft to a common architecture and using common components.
- Opportunities for future mission control systems
 - Develop “power tools” for operators: i.e., new mission planning system.
 - Greater autonomy of spacecraft can produce substantial cost savings. We are beginning to do this with the new generation of spacecraft called “small explorers” that require only one operations team shift per day.

ROLE FOR TECHNOLOGY

- Standards
 - Packet data systems.
 - We need standard ways of representing data products to ease the job of interdisciplinary data analysis. The CCSDS is working on this with their Standard Formatted Data Unit.
- Modern computing machinery
 - We are moving away from large, mainframe computers in favor of distributed systems and custom-designed VLSI hardware processors where appropriate.
 - We need more capable flight computers to achieve higher levels of automation on board.
- Maximizing reuse of software
 - New software environments enable porting software across different vendors’ hardware.
 - Significant reuse will come only if we can make the functions and data structures similar from mission to mission.
- Automation
 - Many tasks in control centers are routine and deterministic: ripe for automation.
 - Expert systems are currently used as advisors to operations personnel — in an off-line sense. It’s time to go further. Examples are Cosmic Background Explorer (COBE), Johnson Space Center mission control, and STGT fault-isolation and recovery.
 - As we gain more experience and confidence, expert systems can be used more directly in the command and control process. In the future, they might be on board the spacecraft, free from the uncertainties of the communications process.

CONCLUSION

I have described the operations services provided by NASA’s Office of Space Communications and discussed our common need for reducing the cost of operations. I have noted some economies we have realized, and explored a number of opportunities for further economies.

The key point is that economies in space operations can be achieved — especially if we change the way we build spacecraft and how they operate.

Standard services and interfaces can be expected to shorten the preparation time for the ground system. A similar benefit should be achieved on the flight element, since there is less “re-invention.”

I suggest that we use this conference to share ideas and experiences in reducing space operations costs, so that our respective space agencies can pursue more exciting and productive missions.

ROLAND IVARNEZ

Head, Division for Geostationary Satellites, Ground Operations
Centre National d'Études Spatiales, Toulouse Space Centre

This paper presents an overview of current space operations performed by the French space agency at the Toulouse Space Centre. The missions are briefly presented and a description of the architecture of the ground facilities involved in the space operations is provided. The improvements subject to current activity are presented and the associated methods and tools are listed.

INTRODUCTION

The Centre National d'Études Spatiales (CNES) is responsible for the implementation of French space policy. It is a government scientific and technical organization run on a commercial and industrial basis.

CNES facilities are sited in the Paris area, at Toulouse, and in French Guyana. The Toulouse Space Centre is the main engineering and technological facility and is responsible for developing and operating French space missions.

The Toulouse Space Centre is in charge of operating scientific and application space systems covering four domains of activity:

- Earth observation satellites
- Geostationary telecommunication satellites
- Mobile search and rescue
- Science

A ground network and multimission facilities are operated to support these activities.

EARTH OBSERVATION SATELLITES

The SPOT concept is a standard spacecraft that can accommodate various payloads for Earth observation. The main payload comprises two high-resolution visible radiometers.

The SPOT system has been operated by CNES since February 1986. The space segment comprises the SPOT 1 and SPOT 2 satellites. The SPOT 3 satellite will be launched in September 1993. A new family of SPOT satellites is under development and, with SPOT 4, will extend the Earth observation mission.

The SPOT satellite orbit is polar and Sun-synchronous, with an altitude of 832 km. This orbit is maintained in such a way that the satellite local time is always the same, allowing for constant lighting conditions.

The ground segment for SPOT includes the following:

- The telemetry, tracking, and command (TT&C) S-band ground station network, comprising the CNES S-band network complemented by the Kiruna (Sweden) station.
- Image receiving stations associated with image processing centres.
- The Mission and Control Centre responsible for satellite monitoring and control, orbit control, and management of the payload work schedule.

GEOSTATIONARY TELECOMMUNICATION SATELLITES

MISSIONS

The first Franco-German geostationary experimental satellites, Symphonie A and B, were launched in 1974. Since then, three major programs have been pursued:

- Telecom 1A, 1B, and 1C
- Telecom 2A and 2B
- TDF1 and TDF2

The Telecom system comprises the Telecom 1C, the Telecom 2A and 2B satellites, and the associated ground control facilities. Since 1984, this system has performed two main missions: business communications service and telephone and television service.

The business communications service uses the 14-/12-GHz band. The Telecom 1C satellite provides six transponders, while each Telecom 2 satellite provides 11 transponders. The telephone and television service establishes links between the French mainland and its overseas territories, using the 6-/4-GHz band. The Telecom 1C satellite provides four transponders, while each Telecom 2 satellite provides five transponders.

France Telecom is in charge of development and operation of the French public telecommunications network. On behalf of France Telecom, CNES is responsible for the control of the satellite.

The TDF1 and TDF2 satellites are part of a Direct Broadcast Satellite family developed cooperatively by Germany and France under the responsibility of TéléDiffusion de France (TDF). TDF is responsible for broadcasting and transmission for public and private television and radio stations. Collocated at 18.8° West, the TDF satellites operate at 18/12 GHz and provide four channels for high-power TV broadcasting.

Concerning the operation of the national satellites, the Toulouse Space Centre is in charge of

- Developing, testing, and running the ground control facilities.
- Performing the station positioning operations.
- Performing the stationkeeping operations.

The Toulouse Space Centre also provides station positioning services for external entities.

STATION POSITIONING

Since the launch of Symphonie A in 1974, CNES has successfully brought 14 operational geostationary satellites onto station. Peak activity

occurred between December 1991 and September 1992, when six satellites were put on station. The 14 satellites are listed below.

Telecom 1A	August 1984
Telecom 1B	May 1985
TDF 1	March 1988
TELE-X	October 1988
Telecom 1C	April 1989
TDF 2	July 1990
Inmarsat 2 F1	October 1990
Inmarsat 2 F2	March 1991
Telecom 2A	December 1991
Inmarsat 2 F3	December 1991
Arabsat 1C	February 1992
Telecom 2B	April 1992
Inmarsat 2 F4	April 1992
Hispasat 1A	September 1992

Station positioning operations are planned for Hispasat 1B in April 1993, Turksat 1A in December 1993, and Turksat 1B in December 1994.

Station positioning operations are characterized by

- Short duration (less than one month).
- Use of a worldwide network.
- Involvement of many entities.

The typical ground segment architecture for station positioning is illustrated in Figure 1.

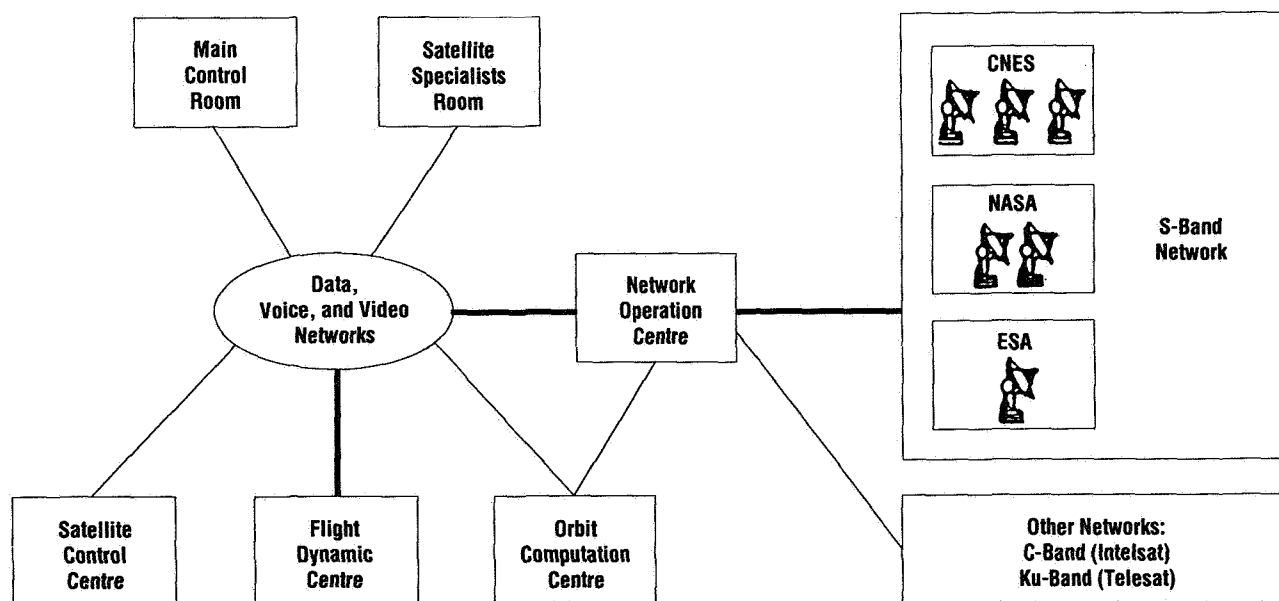


Figure 1. Satellite Positioning Ground Architecture.

The functions of the ground facilities involved in satellite positioning are as follows:

- The Main Control Room provides access to information produced by the other entities that is necessary for coordinating and managing the operations.
- The Satellite Specialists Room hosts satellite manufacturer staff responsible for monitoring the various satellite subsystems.
- The Satellite Control Centre performs satellite monitoring and commanding according to flight control procedures.
- The Flight Dynamics Centre computes and displays the orbit parameters, the satellite attitude, and the manoeuvres parameters.
- The Orbit Computation Centre (OCC) computes the satellite's orbit and provides pointing data to the ground stations.
- The Network Operation Centre (NOC) provides the interconnection with the supporting network. This network can be either the CNES S-band network complemented by ground stations from the European Space Agency (ESA) and/or the National Aeronautics and Space Administration (NASA), or a C-band or K-band external network.

STATIONKEEPING

The control of national geostationary satellites involves facilities dedicated to each satellite family. The typical ground control architecture is presented in Figure 2. The organization of

CNES ground facilities for geostationary satellite stationkeeping is described below.

Dedicated ground stations provide the telemetry and telecommand links with the satellites, using the mission frequencies (C-band for the Telecom 1/Telecom 2 system; K-band for the TDF system). The stations provide the measurements for orbit computation ("turn-around" two-way ranging for the Telecom system; range and angle measurements for the TDF system). The CNES S-band network is used for range and angular measurement calibration or in case of satellite attitude loss.

A data transmission system interconnects the various elements of the ground system.

The Satellite Control Centre fulfills the following functions:

- Telemetry real-time monitoring and control.
- Telecommands preparation and transmission.
- Telemetry off-line processing (trending).
- Ground station remote monitoring and control.
- Orbit and manoeuvre computation.

These functions are distributed on computer systems (shown in Figure 2).

MOBILE SEARCH AND RESCUE

The COSPAS-SARSAT program provides satellite aid to search and rescue operations for maritime, aeronautical, and terrestrial vehicles anywhere in the world. The program interfaces permanently with the International Civil Aviation Organization and the International Maritime Organization.

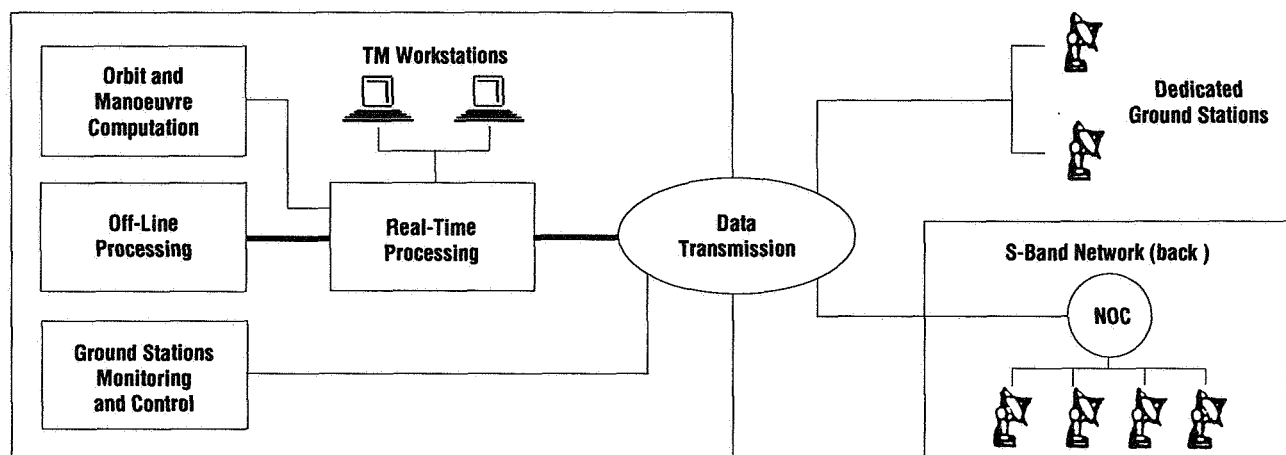


Figure 2. Satellite Control Centre Architecture.

The nominal configuration comprises four satellites relaying signals from distressed units. A local user terminal (LUT) receives the data from the satellites, calculates the position of the activated distress beacon, and transmits the results to the Mission Control Centre (MCC).

The French MCC is operated at the Toulouse Space Centre by a joint team comprising CNES and civil aviation personnel. Its main functions are to collect, store, and sort the data from LUTs, to ensure the exchange within the COSPAS-SARSAT system, and to distribute alert and location data to associated rescue coordination centres.

SCIENCE

The CNES Toulouse Space Centre is responsible for the preparation and operation of French experiments flying on board scientific satellites, as well as the acquisition, preprocessing, and distribution of related data to the scientists.

GROUND NETWORK AND MULTIMISSION FACILITIES

To support space missions, the Toulouse Space Centre operates the CNES S-band network. The function of the network is to provide telemetry and telecommand links with the satellite in orbit. It also performs range and range rate measurements necessary for orbit determination.

The CNES S-band network comprises the following elements:

- Three ground stations located at Aussaguel (France), Kourou (French Guyana), and Hartbeesthoek (South Africa).
- The data transmission system.
- The OCC, which computes the orbit of the supported satellite and provides pointing data to the ground stations.
- The NOC, which monitors and controls all the elements of the network.

The CNES S-band network can be extended with ground stations or ground networks belonging to other entities. ESA and/or NASA S-band

ground stations are used during station positioning operations when performed in S-band. The CNES data communications system is fully compatible with the NASA network and a gateway was developed that allows the interoperability of CNES and ESA networks. The Intelsat C-band network was used during the Inmarsat 2 and Arabsat 1C satellites station positioning.

Multimission facilities are operated for the benefit of the missions. These facilities include

- Time distribution
- Video distribution network
- Voice conferences
- Main Control Rooms
- Satellite Specialists Rooms

IMPROVEMENT OF OPERATIONS EFFICIENCY

Besides the development of new ground control facilities (e.g., SPOT 4), emphasis is placed on the improvement of operations efficiency. The goals are to

- Minimize the number and the effects of human errors.
- Improve the capability to react properly to contingency situations.
- Improve knowledge of satellite behavior.
- Improve productivity.

To this end, critical analyses of the organization, the documentation, and the procedures of the various operational entities are conducted by internal and external audits. New tools have been developed or are under development to support these objectives, such as

- Computer-aided operational documentation production.
- Computer-aided report production.
- Satellite simulators used for satellite procedures validation and for training.
- An expert system to be used within the Telecom 2 control centre.
- Telemetry monitoring and processing on personal computers.

SESSION A

FUTURE MISSIONS AND OPERATIONS

CO-CHAIRPERSONS

J. H. Kwok, JPL, California Institute of Technology, USA

J. F. Muratore, NASA Johnson Space Center, USA

L. J. Uljon, NASA Johnson Space Center, USA

Summary	31
J. H. Kwok	

A.1 Common Front End Systems for Space Shuttle and Space Station Control Centers at Johnson Space Center	33
L. Uljon and J. Muratore, NASA Johnson Space Center, Houston, Texas, USA	

A.2 Mission Operations Update for the Restructured Earth Observing System (EOS) Mission	39
A. C. Kelly and E. S. Chang, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA	

A.3 ASTER, a Multinational Earth Observing Concept	45
G. W. Bothwell, G. N. Geller, S. A. Larson, A. D. Morrison, and D. A. Nichols, JPL, California Institute of Technology, Pasadena, California, USA	

A.4 Ground Operation of Robotics on Space Station Freedom	51
Z. A. Wojcik and D. G. Hunter, Canadian Space Agency, Vanier, Ontario, Canada; M. R. Cantin, KEOPS Informatique, Montréal, Québec, Canada	

A.5 Mission Planning for Shuttle Imaging Radar-C (SIR-C) with a Real-Time Interactive Planning Software	57
S. K. Potts, JPL, California Institute of Technology, Pasadena, California, USA	

A.6 SAMPEX Payload Operation Control Center Implementation	63
D. Mandl, J. Koslosky, R. Mahmot, and M. Rackley, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA; J. Lauderdale, Computer Science Corporation, Laurel, Maryland, USA	

A.7 A User View of the EURECA Ground Segment	69
P. Ferri and S. J. Kellock, ESA European Space Operations Centre, Darmstadt, Germany	

A.8 Development of the Cassini Ground Data System in a Multimission Environment	75
G. Madrid and G. Wanczuk, JPL, California Institute of Technology, Pasadena, California, USA	

A.9 SPOT Satellite Family: Past, Present, and Future of the Operations in the Mission and Control Center	81
P. Pacholczyk, CNES Toulouse Space Centre, France	

SUMMARY — SESSION A

J. H. KWOK
Jet Propulsion Laboratory

There were 10 papers accepted for this session, which was co-chaired by Linda Uljon from NASA Johnson Space Center. John Muratore, the original co-chair, could not attend the symposium because of other commitments with an upcoming Shuttle launch.

One paper ("Overview of Tropical Rainfall Measuring Mission Data System," by D. Han and L. Brown of NASA Goddard Space Flight Center) had to be withdrawn because of lack of travel funds; I was notified in advance.

I counted over 100 attendees in the first two presentations; this number had diminished to about 45 at the end of the session.

Overall, I thought the turnout was really good. Three international papers were presented, from France, Canada, and Germany. The facility and the support personnel were first-rate. The only complaint I had was that the room was too long for the projected image, so people who sat further than one-third of the length from the front had a difficult time seeing the viewgraphs.

The trends I detected from the papers given in this session are:

- There will be increasing cooperation among nations, to the extent that mission operations and payload operations centers may be located in different countries.
- I was impressed by the high data rates (hundreds of Mbits per second) and large amount of data volume to be handled, analyzed, and archived in future missions. The spacefaring nations need to work on how we handle data traffic and data sharing in the future.
- I was pleased with the increased use of the CCSDS standard, the thinking going on in reusable ground systems, multimission support, and evolution of current systems.

I am pleased to have been part of SPACEOPS 92.

PRECEDING PAGE BLANK NOT FILMED

N94-23838

COMMON FRONT END SYSTEMS FOR SPACE SHUTTLE AND SPACE STATION CONTROL CENTERS AT JOHNSON SPACE CENTER

Linda Uljon and John Muratore

NASA, Johnson Space Center
Houston, Texas

ABSTRACT

In the beginning of the fiscal year 1992, the development organizations of Johnson Space Center (JSC) were poised to begin two major projects: the Space Station Control Center and the refurbishment of the telemetry processing area of the Space Shuttle Mission Control Center. A study team established that a common front end concept could be used and could reduce development costs for both projects. A standard processor was defined to support most of the front end functions of both control centers and supports a consolidation of control positions which effectively reduces operations cost. This paper defines that common concept and describes the progress that has been made in development of the Consolidated Communications Facility (CCF) during the past year.

mission control, ground control, telemetry processing, space station, space shuttle

1 BACKGROUND

In preparation for ground support of the upcoming space station, Johnson Space Center had begun to plan a construct a Space Station Control Center (SSCC). A building adjacent to the Mission Control Center (MCC) was in construction and requirements had been written. At the same time, development efforts in the MCC had

been refocused to replacement of the oldest and least reliable machines, those of the front end. A study team was established to determine if there was merit in a joint development effort of a common design. It was hoped that such an effort would reduce both development and recurring costs.

1.1 Three-Layer Model

The study began by defining a model of the front end functions. A three layer view of the processing was produced and is diagrammed below.

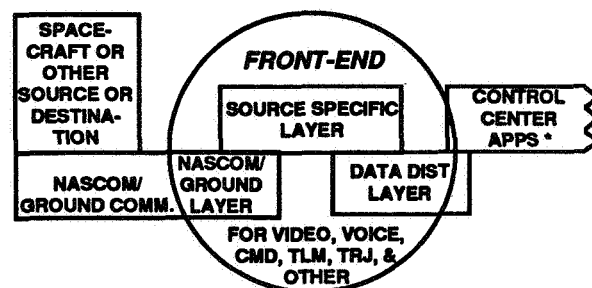


Figure 1. Three layer view of the front end.

Layer 1 is that processing which terminated the ground-to-ground protocol and interfaced with the external world. The Goddard Space Flight Center (GSFC) communications network protocol is handled in this layer, thereby isolating the remainder of the control center from any changes in ground network.

Layer 2 provides the source specific processing for the front end. This layer does the specific format manipulation based on variations in spacecraft communications. It, in turn, isolates any changes in spacecraft telemetry from any other areas of the control center.

The final layer, layer 3, formats the data so that it can be distributed to the remainder of the control center. Ideally, the format is identical despite the spacecraft. This allows common applications, such as display building programs, to use and understand the data independent of source.

Each processing layer removes data formatting variation such that the entire front end produces a set of data which is identical despite its source. An application can use the data from any spacecraft without modification to its processing code. This provides an excellent opportunity for duplication of several applications downstream of the front-end processing. Additionally, future support for additional spacecraft becomes easier.

1.2 Layers vs. Requirements

This model provided an excellent framework against which to map the requirements of the shuttle and space station programs. In the mapping of these functions, layer 1 was designated the institutional layer. It contained the demultiplexers and modems that receive the data from external world. It also included the recording, switching and routing functions. It included the processing to handle the ground network data which included data from other NASA sites, network scheduling and network status data processing. The requirements for the space station and shuttle data processing revealed that the layer 1

functions were identical for both programs.

Layer 2 isolated most of the differences between the spacecraft telemetry. The space station telemetry consists of CCSDS packets using Reed-Solomon encoding. Its telemetry object lists are nonperiodic. The shuttle telemetry could require decryption, and utilizes Inter-Range Instrumentation Group (IRIG) standards. Its periodic nature is key to validating and decommutating the data. It became clear in mapping the requirements for this layer that the two programs were completely different. However, if the unique processing could be isolated, it appeared that a synergistic approach to the front end could be achieved. Spacecraft unique processing of the data could be performed on a single type of machine. Using a single platform, even with a different configuration of boards, would allow a synergistic approach to the maintenance and operations of the machines.

Once the data has been extracted from its down link format, layer 3 provides distribution of that data to the other areas of the control center. Development of a common data format provides the potential that the data acquisition functions used to receive the data throughout the control center can be identical and additional synergistic savings can be created. The code to perform layer 3 functions for the shuttle telemetry differ somewhat from that of the space station program, but both sets of code can easily run on the same platform.

The diagrams below illustrate the allocation of the control center requirements to the three-layer model.

INTEGRATED FRONT-END PROCESSOR (D/L)

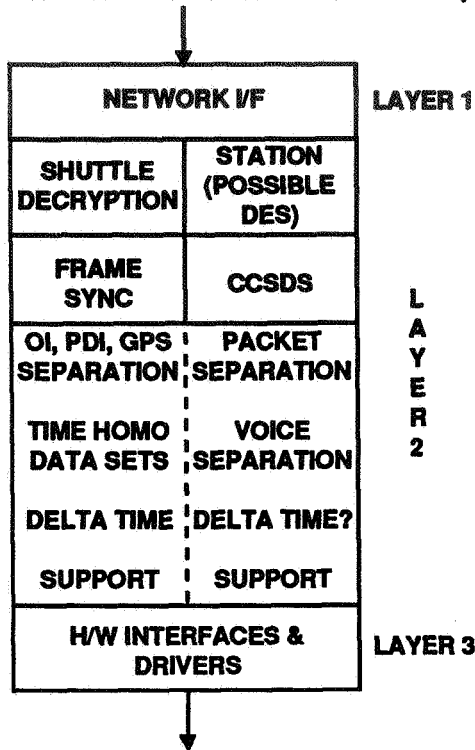


Figure 2. Mapping of downlink front end processing.

Correspondingly, the uplink function was mapped in reverse--building outbound commands in a layered fashion. The basic elements of these layers are given in the Figure 3. Once again, it is apparent that the functions are parallel and that the programs could benefit from synergism.

1.3 Considering the Marketplace

The study team evaluated the requirements and did an initial survey of the marketplace. A straw-man architecture in which the front-end processor was a VME-based machine was proposed. Specialized boards to do the spacecraft-unique processing were found to be available that would run on such a platform. An estimate for a common front end for both control centers at

INTEGRATED FRONT-END PROCESSOR (U/L)

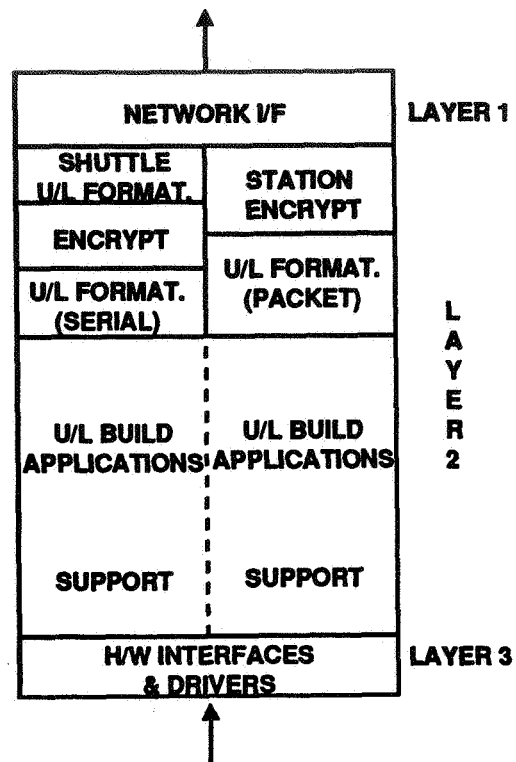


Figure 3. Mapping of uplink front end processing.

JSC determined that NASA could save 15% of the estimated development costs and reduce the maintenance staff by 30. Some savings result from the use of a single piece of hardware for both programs. Some savings is due to the single development effort on common platforms customized for each program. The results of this study were used to launch the Consolidated Communications Facility (CCF) project. This was approximately one year ago.

2. CURRENT STATUS

Since that time, the requirements for the consolidated system have been written and the project has been divided into 5 subsystems. These were: the Consolidated Data Select Switch (CDSS), the Consolidated Data Recording Subsystem (CDRS), the

Front End Processor Subsystem (FEPS), the Consolidated Distribution Subsystem (CDS) and the Test and Checkout Subsystem (TCS). Each of these subsystems has been designed and procurements are in progress. A mapping of the subsystems to the layered concept is shown below.

2.1 Consolidated Data Select Switch (CDSS)

The CDSS consists of high-rate and low-rate data switching, accepting multiple data lines from external sites and routing data to external sites. The low rate data switch

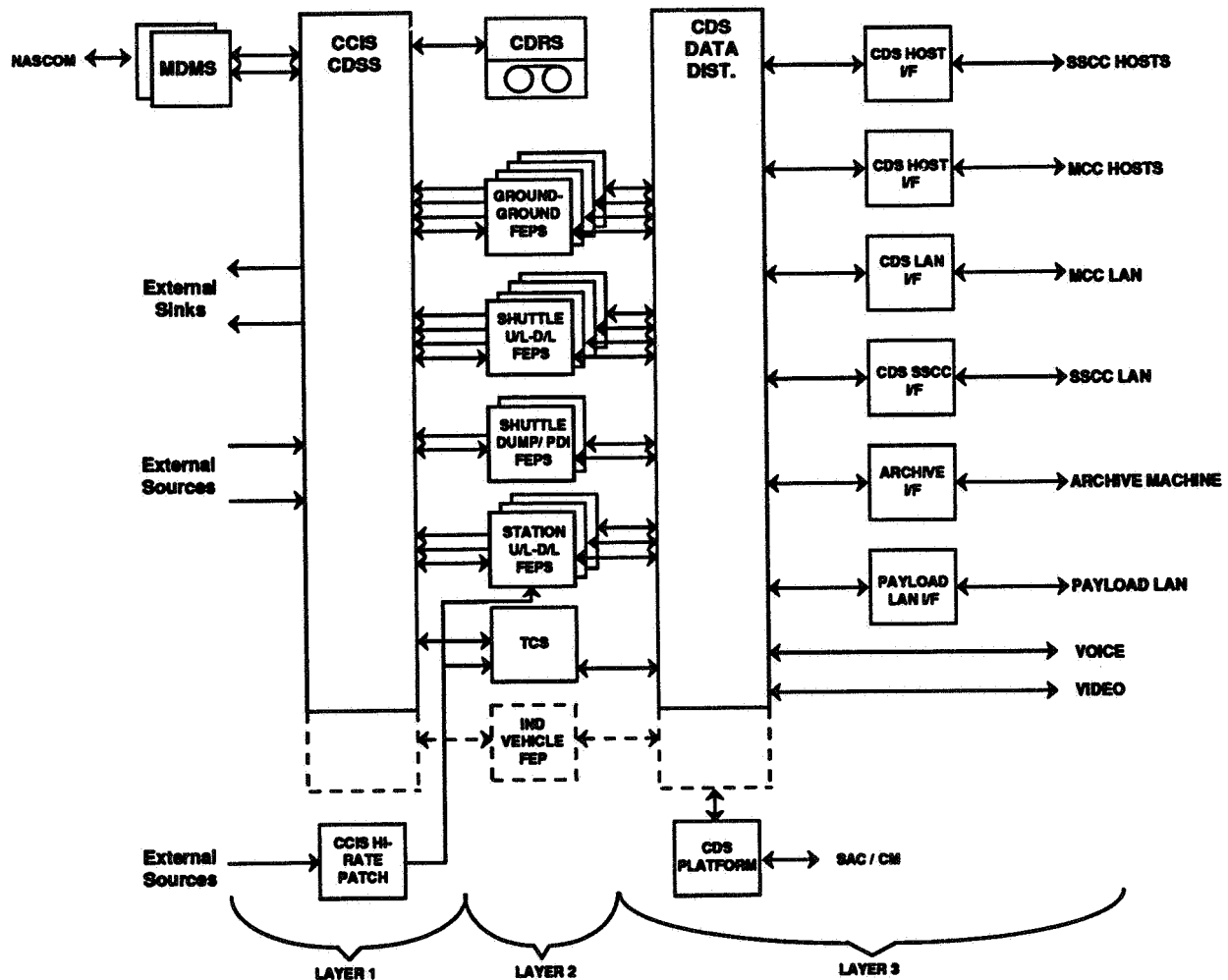


Figure 4. A mapping of the CCF components to the three-layer model.

handles the majority of this work--it is currently specified as a 600x500 port switch. The Request For Proposal (RFP) for this switch is being developed. The high rate data has few lines and will be patched rather than switched.

2.2 Consolidated Data Recording Subsystem (CDRS)

The planned CDRS technology is borrowed from the television industry which keeps commercials on cassettes. A robot then queues the tapes automatically and plays them as scripted. The control center taping system is envisioned as two sets of robotically operated tape silos. Since recorders do not operate efficiently at the slow data rates of the control center, data will be buffered and sent to the recorders in bursts. The recording system is expected to be run with minimal operator intervention. The operator will be needed only to remove any tapes for permanent storage or to arbitrate playback capability. This approach will reduce operations costs--a strong theme in the development of the CCF.

2.3 Front End Processing Subsystem (FEPS)

The FEPS contains the bulk of the processing in the CCF. The current design of the Mission Control Center contains several specialized hardware devices designed in the sixties and seventies. Over the years, other systems based on more modern COTS technologies have partially replaced the front end processing capabilities, but never were able to completely do so. The result is that the Mission Control Center now has four different design approaches to the telemetry processing. All will be replaced with the CCF.

The design of the CCF FEPS is based on a

single VME-based machine. The machines will be tailored to suit the specifics of the spacecraft data for which it is used with special processing boards and software. The FEPS do all of the layer 2 processing. This isolation of the layer 2 to a single processor means that the CCF could be expanded to handle future vehicles with change only in the FEP processing. Additionally, the FEPS provide the processing platform to create the common data format discussed in layer 3. The procurement for the FEPS is currently in progress; replies to the RFP are due in January.

2.4 Test and Checkout Subsystem (TCS)

The TCS will run on the FEPS platform and will provide the ability to run test data streams through the system and score the output to assure proper processing. The TCS will be run routinely to validate the hardware and software is ready. It will also be used by maintenance personnel to locate problems.

2.5 Consolidated Distribution Subsystem

The CDS provides distribution of the data to the computation and display systems throughout the control center complex, including the older IBM mainframes and the newer UNIX workstations. Because the format of the data is common in nature for both programs, a single display processor can be used for either shuttle or space station data. This basis for the front end output simplifies the consolidation of the remainder of the control centers for each program.

3 SUMMARY

By looking at ground support for a joint space station and shuttle project, NASA has found a better and cheaper way of provid-

ing a telemetry processing area. The definition of a front end concept and mapping the functions required by both the shuttle and space station programs clarified the needs of each program. From this, NASA was able to determine what common hardware and software could support both. The subsystems were defined and designs done to accommodate the needs of both programs, as well as expand to future. The procurement of much of the CCF is currently in progress and the entire project is expected to be complete in 1996.

MISSION OPERATIONS UPDATE FOR THE RESTRUCTURED EARTH OBSERVING SYSTEM (EOS) MISSION

Angelita Castro Kelly *
Edward S. Chang +
EOS Project
NASA/Goddard Space Flight Center
Greenbelt, Md U.S.A. 20771

N 9 4 - 2 3 8 3 9

1.0 INTRODUCTION

The National Aeronautics and Space Administration's (NASA) Earth Observing System (EOS) will provide a comprehensive, long term set of observations of the Earth to the Earth science research community. The data will aid in determining global changes caused both naturally and through human interaction. Understanding man's impact on the global environment will allow sound policy decisions to be made to protect our future.

EOS is a major component of the Mission to Planet Earth program, which is NASA's contribution to the U.S. Global Change Research Program. EOS consists of numerous instruments on multiple spacecraft and a distributed ground system. The EOS Data and Information System (EOSDIS) is the major ground system developed to support EOS. The EOSDIS will provide EOS spacecraft command and control, data processing, product generation, and data archival and distribution services for EOS spacecraft. Data from EOS instruments on other Earth science missions [e.g., Tropical Rainfall Measuring Mission (TRMM)] will also be processed, distributed, and archived in EOSDIS. The U.S. and various International Partners (IP) [e.g., the European Space Agency (ESA), the Ministry of International Trade and Industry (MITI) of Japan, and the Canadian Space Agency (CSA)] participate in and contribute to the international EOS program. The EOSDIS will also archive processed data from other designated NASA Earth science missions (e.g., UARS) that are under the broad umbrella of Mission to Planet Earth.

2.0 EOS RESTRUCTURING

Various events have resulted in the restructuring of the EOS program over the past year. The

Congressional Committees on Appropriation directed NASA to restructure EOS based on three principal reasons: to focus the science objectives of EOS on global climate change, considered the most critical problem of global change; to increase the resiliency and flexibility of EOS by flying instruments on multiple smaller spacecraft rather than large platforms; and, to reduce the cost of the overall program. NASA, through internal and external reviews, discussions with the science community, and numerous engineering studies of spacecraft configurations and launch options, restructured EOS to meet congressional constraints and still satisfy the needs of the scientists investigating global climate change.

In addition to the above, NASA programmatic decisions resulted in an expanded scope for the EOSDIS. Original plans utilized two future NASA institutional elements: the Customer Data and Operations System (CDOS) and an upgrade version of the NASA Communications System (NASCOM II). These elements would provide the forward and return link processing services and mission critical communications functions for EOS and future NASA missions with high data rates. In the summer of 1991, NASA Headquarters transferred functional responsibilities for these future elements to the EOS Program Office; thus the EOS Data and Operations System (EDOS) and the EOS Communications System (ECOM) have been incorporated within EOSDIS.

In the spring of 1992, EOS was directed to include the Landsat-7 mission under the EOS umbrella. Landsat-7 is being procured under a joint Air Force/NASA contract. This includes the flight operations segment (operations control center) and the science processing software which will be used in the Landsat-7 ground processing system, the latter to be procured under a separate contract. The Landsat-7 operations control center will be physically located in the EOSDIS building at GSFC. Data archival and distribution will be provided under EOSDIS.

* EOS Mission Operations Manager (MOM)
+ AM Spacecraft Project Operations Manager

3.0 EOS SPACECRAFT CHANGES

The major differences between the original and restructured spacecraft configuration are highlighted in Table 1.

the increase in the number of on-orbit spacecraft).

4.1 Forward and Return Link Operations

Figure 1 on the next page provides an overview of the current EOS mission concept. EDOS and

CHARACTERISTICS	ORIGINAL CONFIGURATION	RESTRUCTURED CONFIGURATION
Number of Series/Total Number of Spacecraft	2/6	5/17
Nodal Crossing Time	EOS-A: 1:30 PM ascending EOS-B: TBD	EOS AM: 10:30 AM descending EOS PM: 1:30 PM ascending EOS CHEM, ALT, AERO: TBD
Number of Instruments per Spacecraft	EOS A: 15 EOS B: approx. 14	EOS AM: 5 EOS PM: 5 to 6 EOS AERO: 1 EOS CHEM: 4 EOS ALT: 3
Launch Vehicle	Titan IV	EOS AM, PM, CHEM: ATLAS IIAS class EOS ALT: Delta class EOS AERO: Scout class or Pegasus
Forward Link Data Rate	1 to 100 Kbps	1 to 10 Kbps
Return Link Data Rate	up to 300 Mbps	up to 150 Mbps
Spacecraft Design	Common design and modularity of subsystem components	Commonality within series

Table 1 - Original vs. Restructured Spacecraft Configuration

The original EOS spacecraft configuration consisted of two series of large platforms, EOS-A and EOS-B. Each series had a large number of instruments and three spacecraft per series, with each spacecraft having a five-year lifetime and an identical instrument complement.

The restructured EOS spacecraft configuration consists of five series of spacecraft, each with a different flight configuration based on scientific measurement objectives. Table 2 summarizes the EOS series and planned launch dates.

4.0 EOS GROUND SEGMENT UPDATE

The EOS ground segment and hence the ground operations concepts have undergone a number of changes as a result of changes to the spacecraft configuration and NASA management decisions. As stated earlier, EOSDIS has been expanded to include two new elements, the EDOS and ECOM. Consequently, mission operations concepts have been revisited in response to the restructured configuration (e.g.,

ECOM provide the interface between the EOSDIS Core System (ECS) and the spacecraft utilizing NASA institutional systems. EOS spacecraft use the Tracking and Data Relay Satellite System (TDRSS) for normal spacecraft operations and science data acquisition. The Deep Space Network (DSN), the Ground Network (GN), and the Wallops Orbital Tracking Station (WOTS) are used for contingency operations. EDOS and ECOM elements are located at White Sands, NM, Fairmont, WV, and the GSFC.

Delivery of mission critical data is the responsibility of ECOM. ECOM will transport forward link data from the EOS Operations Center (EOC) to the EDOS/White Sands facility and from EDOS to the TDRSS Ground Terminals (TGT) for uplink to the spacecraft through the TDRS. ECOM will transport the return link data from the TGTs to EDOS at White Sands, from where realtime housekeeping data will be sent to the EOSDIS control center [i.e., the EOS Operations Center (EOC)] for

U.S. EARTH OBSERVING SYSTEM (EOS)				
Spacecraft Series	No. of Spacecraft	Launches * (Tentative)	Nominal Lifetime (Years)	
			Per Spacecraft	Series
AM	3	June 1998, 2003, & 2008	5	15
PM	3	December 2000, 2005, & 2010	5	15
AERO	5	September 2000, 2003, 2006, 2009, & 2012	3	15
ALT	3	June 2002, 2007, & 2012	5	15
CHEM	3	December 2002, 2007, & 2012	5	15

* Launch dates after AM1 are for planning purposes.

Table 2 - EOS Launch Schedule

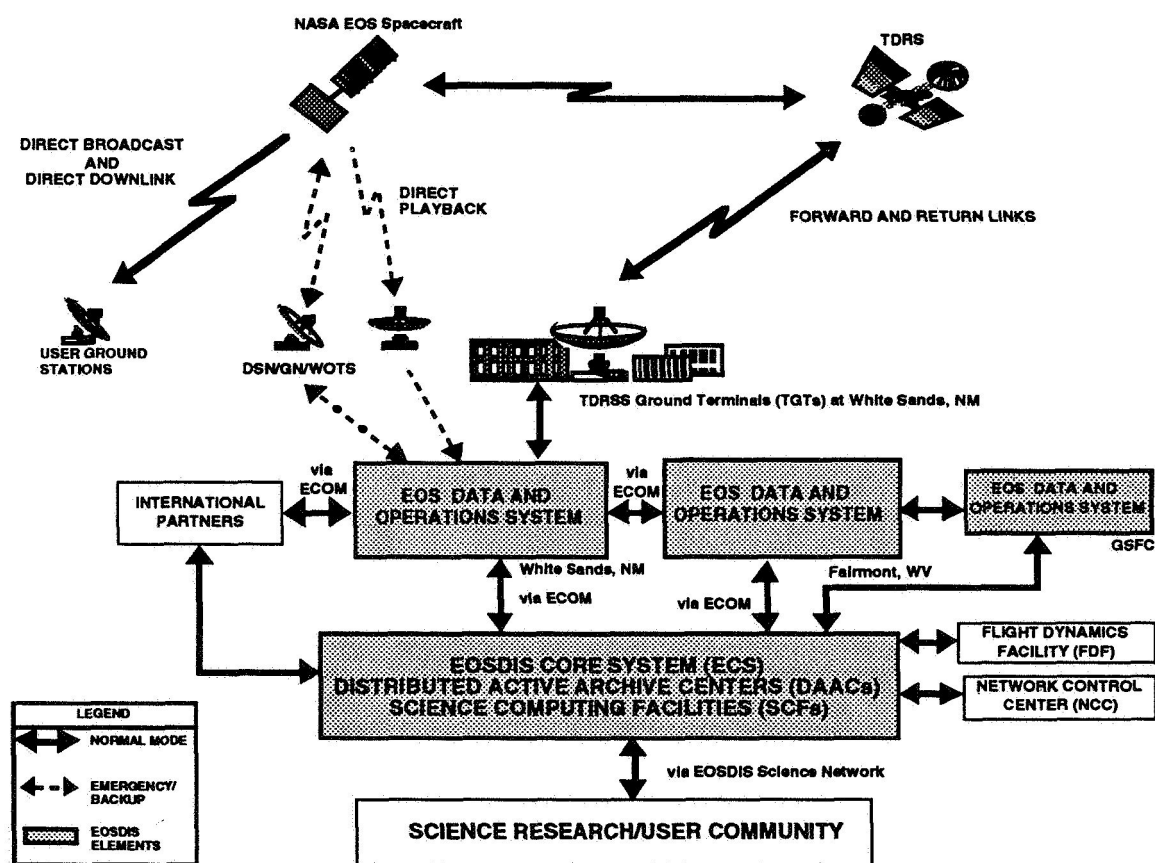


Figure 1 - EOS Mission Concept

health and safety and quicklook analysis; science data will be sent to the EDOS/Fairmont facility for Level 0 production processing. ECOM will transport production data sets to the designated EOSDIS data processing and archive centers or user facility for higher level processing.

5.0 MISSION OPERATIONS

Figure 2 on the next page compares the original operational profile to the restructured profile. During the 15 year plus operational lifetime of EOS, as many as five spacecraft will simultaneously be performing normal operations.

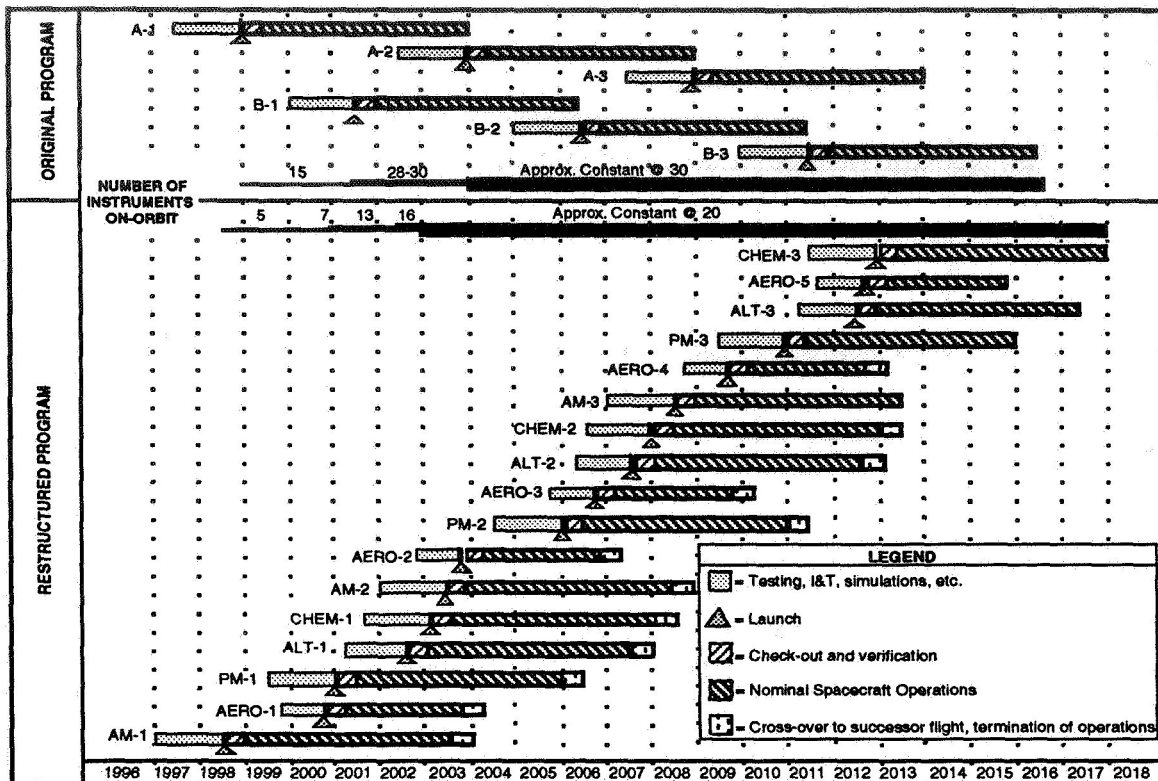


Figure 2 - EOS Operational Profile (Original vs Restructured)

During periods of cross-over operations (3 to 6 months), up to seven spacecraft (e.g., year 2003) will be on-orbit, all requiring planning and scheduling, command and control, and monitoring for health and safety and spacecraft performance. In addition, prelaunch operations (e.g., I&T, simulations, ground system compatibility, and operational readiness tests) for successor spacecraft must also be supported.

To determine the potential physical and personnel resources required to support multiple spacecraft operations, an in-depth analysis will be performed to determine the average and peak support requirements for normal and contingency operations. This analysis will consider setting priorities for scheduling mission support for the various spacecraft, specifying the additional resources required during contingency operations, and determining the requirements for reliability and maintainability of the ground system. The ground system will be "optimized" where possible to provide the necessary capabilities.

The EOC will be developed with the potential to expand and evolve as EOS progresses. The EOC will not require the capability for handling multiple spacecraft until two years after AM1 is

launched. More elaborate planning and scheduling tools may be needed as coordinated operations increase among spacecraft. The EOC will be designed so that needed enhancements and capabilities can be added and tested with minimal impact to ongoing operations. The flight operations systems will also support prelaunch operations for successor spacecraft as well as ground system upgrades and maintenance.

5.1 Spacecraft Command and Control

The original EOSDIS flight operations system consisted of the EOC, Instrument Control Facilities/Instrument Control Centers (ICF/ICC), and Instrument Support Terminals (ISTs). The EOC was responsible for the spacecraft health and safety, integrated mission control, mission planning and scheduling, platform commanding, instrument command support, platform maintenance, and overall mission operations. Two ICFs were planned, each housing several ICCs. The ICCs were primarily responsible for instrument health and safety, planning and scheduling, command generation, and instrument maintenance activities. The ISTs were to be used by the Principal Investigators/Team Leaders' (PI/TL) instrument team to support ICC operations.

The same functions remain in the revised operational concept; however, the functions have been redistributed among the elements. For example, the breakup of large platforms into many spacecraft, in conjunction with better knowledge of each instrument's operational complexity, has resulted in the elimination of the two-ICF concept and an increased role for the EOC in instrument operations support. Instrument operational complexity is judged by the amount of ground operation (i.e., planning and scheduling, command generation) required to perform the normal day-to-day instrument operations. Evaluating the current instrument manifest for all EOS spacecraft, only two to three instruments contain sufficient operational complexity to warrant development of an ICC for instrument operations. The ICCs for these instruments perform essentially the same role as that in the original ICC concept.

The operation of all simple or less complex instruments will be a coordinated effort between the EOC and an IST. The allocation of all aspects of instrument operations (e.g., planning and scheduling, command generation, health and safety and performance monitoring, etc.) will be negotiated for each instrument prior to launch. It is envisioned that instrument health and safety monitoring and command generation will be performed at the EOC with support from the IST. During periods of intensive instrument activities (i.e., instrument checkout, calibration periods, anomaly investigation, and instrument software loads), the instrument teams may take a more active role with the EOC providing support as needed.

5.2 Planning and Scheduling

Policy guidelines and the long term science plan for use of the instruments are set by the Program and Project Scientists and the EOS Investigator Working Group (IWG). The long term plan and guidelines form the basis for planning and scheduling at the EOC, ICCs, and ISTs. Since planning and scheduling for instruments that are not operationally complex will normally be minimal, the EOC will schedule daily activities for noncomplex instruments using a baseline instrument operations schedule. The baseline schedule may be periodically changed or updated by instrument teams through the IST. The daily planning and scheduling of complex instruments will be performed by the ICC in coordination with the EOC. The EOC integrates preliminary spacecraft subsystems and instrument activity schedules to determine the Space Network (SN) resource requirements. Upon completion of

negotiations for SN resources with the Network Control Center (NCC), the EOC integrates detailed activity schedules for the instruments and subsystems into a spacecraft activity schedule used for command generation. Changes to the schedule may be made up to 1 hour prior to execution in response to Targets Of Opportunity (TOOs).

6.0 EOS AM1 UPDATE

As the first in the series of EOS spacecraft, EOS-AM1 is furthest along in development, with subsystem and instrument Preliminary Design Reviews (PDRs) currently in progress and an integrated spacecraft PDR scheduled for the fall of 1993.

The resource demands of other spacecraft using the SN-TDRSS will limit the amount of real time space-to-ground interaction available. Therefore, the EOS-AM1 spacecraft is being designed to require a minimum of ground control and interaction under normal spacecraft operations.

The AM1 spacecraft design incorporates Telemetry Monitoring (TMON) functions similar to those used on current spacecraft (e.g., UARS). Selected telemetry points (settable from the ground) from individual subsystem components are monitored by the spacecraft onboard computer. These points are compared to pre-established thresholds or limits, which are also settable or overridden, if necessary, via ground commands. Should an anomalous condition occur, the spacecraft will identify the likely cause and switch to a redundant path or function within the subsystem. If the anomaly continues, the spacecraft reverts to a degraded mode of operations. Subsystem housekeeping data are downlinked to allow ground-based evaluation of spacecraft performance and to identify and investigate the anomalous condition. Under certain conditions, when an anomaly jeopardizes the spacecraft life, the spacecraft will automatically place itself in a "safe mode," where only critical components are operating to maintain spacecraft viability and safety. After the anomaly has been resolved, the spacecraft can be returned to the normal operations mode via ground command.

AM1 spacecraft operations will normally be pre-planned. The AM1 spacecraft will use the S-band Single Access (SSA) link at 10 Kbps to load subsystem and instrument commands. The commands are uplinked to the spacecraft approximately 24 hours prior to execution.

Instrument activities are in general initiated via time-tagged commands stored in the spacecraft onboard computers and/or command tables loaded within an instrument microprocessor. The science data are normally recorded onboard for later transmission to the ground via the TDRSS (normally two 15-minute dumps per orbit). The AM1 instruments and their data rates are listed in Table 4.

made to accommodate the restructured program. Some of the concepts are addressed in this paper. More detail can be found in Reference 5.

8.0 ACKNOWLEDGEMENTS

This paper includes relevant information on the EOS program contained in project presentations and preliminary design review material on EOS-

Instrument Name	Power Consumption (w)		Data Rate (Kbps)		Science Objectives
	Average	Peak	Average	Peak	
ASTER	495	720	8300.0	89200.0	Provide high resolution images of the land surface and clouds for climatological, hydrological, biological, and geological studies.
CERES (2)	95	145	20.0	20.0	Provide continued long-term measurement of the Earth's radiation budget through observation of short and long wave radiation.
MISR	80	135	3800.0	7800.0	Provide continuous multi-angle imagery of the Earth at wavelengths of 440, 550, 670 and 880 nm.
MODIS	295	295	8200.0	11000.0	Provide measurements of biological and physical processes on a 1 km x 1 km scale with emphasis on the study of oceanic, terrestrial, and atmospheric phenomena.
MOPIIT	250	250	6.0	6.0	Measure carbon monoxide and methane concentrations in the troposphere to enhance knowledge of the lower atmosphere.
Totals for AM1	1215	1545	18326.0	108026.0	

Table 4 - EOS-AM1 Instruments

The AM1 spacecraft provides an alternate source of data retrieval through an X-band Direct Access System (DAS). The DAS provides three services: the Direct Broadcast (DB) mode, which transmits data from the Moderate Resolution Imaging Spectrometer (MODIS) instrument at 15 Mbps; the Direct Downlink (DDL) mode, which transmits data from the Advanced Spaceborne Thermal Emission and Reflection Radiometer (ASTER) at 105 Mbps; and the Direct Playback (DP) mode, which transmits all data recorded on the high data rate recorder(s). The DP mode will be used as a backup mode to recover science data in case of extended or total loss of the TDRSS high rate data link (i.e., loss of the HGA or the loss or permanent curtailment of TDRSS support).

7.0 CONCLUSION

The restructured EOS mission is progressing well in all areas. EOS is a long term mission which will provide a comprehensive data base on global climate change. The first spacecraft, AM1, is in preliminary design review stage. Procurement of the EOSDIS is underway. The impacts of the restructuring on mission operations concepts have been evaluated; adjustments have been

AM1. Special thanks go to the EOSDIS flight operations development personnel headed by Steve Tompkins. Thanks also to Gregg Einfalt for his support in preparing this paper.

9.0 REFERENCES

1. Mission Operations Concepts for the Earth Observing System (EOS), 42nd Congress of the International Astronautical Federation, October, 1991.
2. Report to Congress on the Restructuring of the Earth Observing System (EOS), NASA, March 9, 1992.
3. Update: The Earth Observing System (EOS) Forward & Return Link Data Processing & Communications Services, World Space Congress, September, 1992.
4. AM1 Spacecraft Bus Preliminary Design Review, June 1992.
5. EOS Mission Operations Concept (Preliminary), GSFC/Code 423, November, 1992.

ASTER, A MULTINATIONAL EARTH OBSERVING CONCEPT

Graham W. Bothwell
 Gary N. Geller
 Steven A. Larson
 Andrew D. Morrison
 David A. Nichols

N94-23840

Jet Propulsion Laboratory
 California Institute of Technology
 Pasadena, California

ABSTRACT

The Advanced Spaceborne Thermal Emission and Reflection Radiometer (ASTER) is a facility instrument selected for launch in 1998 on the first in a series of spacecraft for NASA's Earth Observing System (EOS). The ASTER instrument is being sponsored and built in Japan. It is a three telescope, high spatial resolution (15 to 90 meters on the ground depending on the telescope) imaging instrument with 15 spectral bands covering the visible through to the thermal infrared. It will play a significant role within EOS providing geological, biological, and hydrological information necessary for intense study of the Earth.

The operational capabilities for ASTER, including the necessary interfaces and operational collaborations between the U.S. and Japanese participants, are under development. EOS operations are the responsibility of the EOS Project at NASA's Goddard Space Flight Center (GSFC). Although the primary EOS control center is at GSFC, the ASTER control facility will be in Japan.

The planning and scheduling of ASTER operations will be a complex exercise, not only because of the international interfaces but also because the instrument can be programmed to function in a variety of configurations, and because the limited downlink bandwidth dictates that the instrument be used in a selective manner to observe specified targets, not in a continuous survey mode. The long term average data rate for this instrument is approximately 8% of its peak capabilities.

Central to ASTER's long term operations will be a global mapping program, which will readily consume the instrument's background observation time. The more foreground operations, i.e. those with less long term

planning or for limited target areas, fall into several categories. The majority of these are routine monitoring operations, each requiring between 1 and 1,000 ASTER scenes. The greatest degree of planning will be for single sites with coordinated ground or aircraft observations. Some investigators require one-time observations of a site or region of the highest possible quality, necessitating repeated observations. Cloud studies can utilize data acquired in a less-structured/random manner. There are also the usual targets of opportunity.

These various observing requirements necessitate a variety of mission operations and data downlinking capabilities. The most complex will involve planning, acquisition, and analysis of data at a rate allowing immediate decisions on subsequent data acquisitions by this or associated instruments/spacecraft. Some data will be downlinked directly to regional receiving stations rather than via the standard EOS routing. The decision-making process required in these operations will entail close collaboration between U.S. and Japanese teams.

ASTER's scientific utilization is the responsibility of a joint U.S.-Japanese Science Team. There is a Team Leader and a number of Team Members making up the U.S. component of the Science Team in a similar manner to other EOS facility instruments. The authors are associated with the U.S. Team Leader in the development of mission operations and ground data systems on behalf of the U.S. Team.

1. INTRODUCTION

EOS, as part of NASA's Mission to Planet Earth, aims to advance the scientific understanding of the Earth by monitoring the various components of the Earth system on a global scale. This includes the acquisition of a 15-year baseline of remote-sensing measurements and the

development of a large data and information system to provide access to this baseline for scientists studying Earth systems, particularly those focusing on global change. The ASTER Project supports these goals by producing scientifically-useful products that quantify various aspects of the earth system, and by providing access to these products via EOSDIS.

The ASTER Project has its beginnings in both Japan and the U.S. In Japan, the Ministry of International Trade and Industry (MITI), organized government funding of the Intermediate Thermal Infrared Radiometer (ITIR) instrument project for remote sensing data to assist in mineral exploration throughout the world. ITIR was to have five channels in the 8-12 μm . In the U.S. a proposal was submitted to NASA to build the Thermal Infrared Ground Emission Radiometer (TIGER) to make quantitative measurements of emitted radiation from the Earth's surface in the 8-13 μm and 3-5 μm atmospheric windows, at spatial and spectral resolutions appropriate for geological, climatological, hydrological and agricultural studies. NASA headquarters decided on a single instrument built by the Japanese based on the ITIR but with an improved capability to meet the needs of the U.S. TIGER team and the EOS community. Negotiations led to the addition of the Japanese ASTER instrument to the U.S. EOS project and the merging of the Japanese and U.S. science teams.

The ASTER instrument will be flown on the EOS AM1 platform, the first of three polar-orbiting EOS platforms. The Japan Resources Observation Systems Organization (JAROS) is responsible for the design and development of ASTER which is subcontracted by JAROS to NEC, MELCO, Fujitsu and Hitachi. Although the primary EOS control center is at GSFC, the ASTER Instrument Control Center (ICC) will be located in Japan. The ICC will be responsible for final operations decisions and preparation of the instrument commands. Goddard performs the uplink. The international Memorandum Of Understanding and Project Implementation Plans between the U.S. NASA and the MITI of Japan are still in preparation.

The ASTER instrument is comprised of three optical telescopes (radiometers): (1) a visible-near infrared 2-telescope system (VNIR) that is capable of providing stereo data for the production of digital elevation models, (2) a short wave infrared telescope (SWIR), and (3) a thermal infrared telescope (TIR). All three

radiometers can be operated independently and all three are individually pointable. The instrument features high spatial and radiometric (thermal) resolution. The nadir-viewing swath width is 60 km. With its pointing capability, ASTER is capable of viewing any point on Earth every 16 days. Because of its polar orbit, it can view any point above 45 degrees every 7-9 days and any point above 69 degrees, every 3-4 days. It takes 48 days to provide full surface coverage.

The goals for the ASTER instrument are to obtain high-resolution, targeted data in the visible and infrared; a global land map using all three radiometers and stereo data using the nadir and back-viewing telescopes of the VNIR. Such information will be used in studies of surface radiation balance, evaporation and evapotranspiration, vegetation, soils, the hydrogeologic cycle, surface-atmosphere interactions, volcanic processes, sea ice and cloud studies and to produce an earth surface digital elevation model.

Since ASTER is the only high-spatial resolution imager planned for EOS AM1 it will be relied upon to provide surface temperatures, surface emitted and reflected radiances, and digital elevation models at a finer spatial scale than other EOS instruments.

The investigators who will utilize ASTER data include the U.S. and Japanese ASTER Science Teams, the EOS Interdisciplinary Scientists, other EOS scientists, government agencies, and members of the international science community at large. The data is also required by other EOS AM1 instruments for calibration and validation of their data. Researchers will be able to request that specific data be acquired for their studies. Raw and processed data products will be available from a data processing and archiving center.

Considerable planning will be required in order to achieve the ASTER science objectives with the Japanese-provided instrument on a U.S. platform for a joint U.S.-Japanese Science Team. Several meetings have been held by the EOS Project, EOS AM1 Project, and the joint Science Teams to insure complete communication.

2. CAPABILITIES AND CONSTRAINTS

The sun-synchronous 705-km orbit of the EOS AM1 platform that will carry ASTER is characterized by a descending 10:30 am

equator crossing, a 98.5-deg inclination, and a 100-minute period. The duration of one cycle (the length of time from one orbit until the nadir track repeats) is 16 days.

EOS AM1 will carry four instruments in addition to ASTER. They are the Clouds and Earth's Radiant System (CERES), the Multi-Angle Imaging Spectro-Radiometer (MISR), the Moderate-Resolution Imaging Spectroradiometer (MODIS) and the Measurement of Pollution in the Troposphere (MOPITT).

The VNIR acquires data in the wavelengths 0.52 μm to 0.86 μm in three bands. All three view through the nadir telescope and one band, the 0.76 to 0.86 μm band, views through the back looking telescope. The SWIR covers the 1.60 to 2.43 μm wavelengths in six bands. The TIR covers the 8.3 to 11.3 μm in 5 bands. The spatial resolution of the VNIR, SWIR, and TIR are 15, 30 and 90 meters on the ground, respectively. The radiometric resolution is <0.5%, <0.5% to 1.3% and <3K for the three telescopes.

Thermal dissipation, power use and data volume all limit the number and extent of observations. The VNIR and SWIR telescope systems will be capable of up to an average duty cycle of 8%, e.g. they can be operated continuously for up to 16 minutes in one orbit; however, to do so will require that they be turned off in the immediately preceding and following orbits. The TIR will be capable of a 16% duty cycle. If, instead of operating continuously, the radiometers are turned on and off more than once in an orbit, the achievable duty cycle is significantly reduced due to the large operating overhead associated with each observation. Each radiometer has different preparation and calibration cycles that are periodically required. The instrument data rates are 62 Mbps for the VNIR, and 23 and 4.1 Mbps for the SWIR and TIR respectively. The two orbit average data rate allowance is 8.3 Mbps.

The cross-track swath-width is 60 km at nadir. The telescopes are capable of slewing independently providing cross-track coverage of up to 232 km. At this time, it appears that lifetime limitations of the slewing hardware will limit the VNIR and SWIR to 10,000 pointing changes each over the 5 year lifetime of the mission. The science request is for 15,000 to 20,000 pointings. The TIR pointing limitation is 200,000, but this number is required to include repointings for thermal and pointing calibrations before and after each observation. The distance

between successive groundtracks at the equator is 172 km.

Most observations require maximum cloudfree viewing. Observations may be acquired many times before a cloud-free view of a particular area is obtained. The ASTER Science Project is currently studying the probability of acquiring the full land surface cloud-free in the 5 year mission. The results may indicate a requirement to use both long-term and short-term cloud predictions in planning.

3. PLANNING AND OPERATIONS DEVELOPMENT

Science requested observations will fall into several categories: routine monitoring; single sites with coordinated observations; one-time, high-resolution, cloud-free images; and cloud studies. The majority of observations are routine monitoring operations, each requiring between 1 and 1,000 ASTER scenes. The greatest degree of planning will be required for single sites with coordinated observations from the ground, from aircraft or from other EOS AM1 instruments to obtain spatial and DEM observations and for calibration and atmospheric correction. Some investigators require one-time observations of a site or region of the highest possible quality, that may necessitate repeated observations to acquire. Cloud studies can utilize data acquired in a less-structured/random manner.

The Science Team has proposed a basic operating plan for ASTER which allocates 5% of its viewing time for emergency observations (natural disasters, etc. and other ephemeral targets that might include power plant accidents, etc.), 20% for Science Team-requested targets for specific research topics, 50% for Regional multi-temporal Monitoring (regional change), and 25% for Global Mapping, e.g. to observe the entire land surface, cloud-free, at least once during the 5-year mission.

Whatever final scheduling plan is adopted, the planning and scheduling of ASTER operations will be a complex exercise, not only because of the international interfaces and the breadth of the science data expected, but also because the instrument can be programmed to function in a variety of configurations, and because the operations constraints will dictate to a large degree the instrument use. The challenge will be distributed between the U.S. and Japan.

The steps by which science data is translated from the wishes of an investigator to data that is specifically useful to his or her investigation, can be divided into uplink and downlink elements.

3.1 Uplink

The uplink elements of the instrument operations begin with the request by an investigator or a Science Team for data acquisition and are completed when the implementing command sequence is transmitted to the instrument from the spacecraft computer.

Operations planning for the uplinking of requests will begin with the design of the data acquisition request (DAR). DARs will have to contain sufficient information to allow them to be evaluated for their scientific merit and appropriateness for the mission and for technical feasibility and, most importantly, to allow them to be ranked for the purpose of scheduling among all the DARs submitted to the system.

The plan for managing data acquisition requests by investigators is not finalized. A preliminary scenario is that any investigator who wants to specify targets and data acquisition parameters will prepare a data acquisition request (DAR) and submit it to the EOS Information Management System (IMS) for preliminary evaluation. (Fig. 1) This will include screening for conflicts with known operational constraints and a search to determine whether the data already exists in the archive. The DAR would be forwarded to the ICC for reformatting into detailed observation schedules and commands. A subset of the ICC and the joint Science Team, using criteria developed with the Project Scientist, will evaluate and prioritize the requests and resolve any conflicts that arise. The set of ranking criteria may include the number of requesters for the same data, the priority of the science, the permanence and changing nature of the target, target size, telescope required, repointing requirements, day or night observation, conflicts, etc. The ICC will be responsible for the final instrument schedule, instrument commanding and the health and safety of the ASTER instrument.

A conflict-free prioritized activities list will be sent to the EOS Operations Center (EOC) at GSFC for creation of a composite schedule and uplink to the spacecraft. Command sequences are stored in the spacecraft memory before being sent to the ASTER instrument.

DARs may be long term, i.e. submitted well before launch of the 5-year mission; short term, submitted 7-21 days prior to the requested observation; or 'targets of opportunity' (TOOs), that may be requested only hours before the target orbits. TOOs may be requests for data on important ephemeral processes, natural or otherwise, including volcanoes and forest fires. Short term requests will be evaluated at the ICC to determine their priority relative to the long term planned observations. If a short term request or a TOO is inserted into the queue, other observations will be bumped and initiate a 'domino process' to re-select all the previously scheduled acquisitions according to their priorities and within the constraints of the instrument and spacecraft. There will be an estimated 350 observations per week.

The ICC and the IMS receive from the EOC the platform composite activity schedule. Members of the science community can access the IMS to obtain status updates for their own requested observations.

Central to ASTER's long term operations will be the global mapping program that has the objective of mapping the entire surface of the planet with all three telescopes. In order to achieve this objective, global mapping data will be collected whenever the telescopes are turned on, but not acquiring specific investigator requested data. At a 40-50% observation rate for global mapping, and assuming 10% cloud-free observations, it could take 4 years to acquire a high percentage of the global map. This might be reduced by strategic planning of the mapping of the requested targets.

The hierarchy of priorities (from the draft MOU between NASA and MITI concerning the flight of ASTER on EOS AM1) that determine the final activity schedule at the ICC and the EOC in descending order are:

1. platform health, instrument health and safety, and data to assist in declared national or international environmental emergency;
2. calibration/validation including special observations to enable cross calibration of instruments, calibration of individual instruments, and support of specific validation measurements;
3. large data acquisitions either to continue long-term data records or acquisitions of time-critical data on specific earth phenomena ;
4. support of large-scale multi-investigator field experiments;

5. smaller data acquisitions including specific requests by International Partners or NASA.

3.2 Downlink

The downlink procedure begins with the transmission of the data from the spacecraft recorder via TDRSS, to White Sands, N. M. (although in some cases it might be direct down-linked to Japan). (Fig. 2) Then it is shipped via EOS Communication System (ECOM) to the EOS Data Operations System (EDOS) in West Virginia where the level 0 data are constructed. In the current baseline plan the level 0 data are then shipped to Japan where level 1 processing is carried out. The Project is evaluating the option of a parallel level 1 processing capability in the U. S. A copy of the level 1 data is shipped from Japan to the Land Processes Data Active Archive Center (DAAC) at the EROS Data Center (EDC) in South Dakota where the data is archived and where it will be processed applying investigator developed algorithms to produce the data products required by each investigator.

The joint U.S.-Japanese Science Team has the responsibility of guiding the development of ASTER data systems including include defining and specifying data products, developing the algorithms and production software to produce the data products from the ASTER data and related data sources, validation and quality assurance of the data products, and finally, software maintenance and upgrading.

The preparation of standard and special data products lists is in process at this time. EOS system constraints have pared down the original list of ASTER data products requested by the Science Team to 6. Algorithms to convert the ASTER observations and related data to useful science products will be developed by members of the Science Team with support from ASTER Science Project algorithm developers. The Project Software development team will translate the algorithms into EOS system compatible production software for final science product preparation by the ASTER Distributed Active Archive Data Center (DAAC).

Requests for data products may be part of a DAR when an investigator initiates a request for new data, or may be made directly to the DAAC for existing data.

3.3 Tools

Tools including a data acquisition simulation tool, a planning and visualization tool, and an evaluation and prioritization decision tool will all be required to provide support to the uplink and downlink development.

With support from the EOS Instrument Operations Support Task, a data acquisition simulation tool is under development that has the capability of generating representative single-cycle targetted and global mapping observations, the '16-Day Scenario'. Scenarios generated by the simulation tool will be useful for long-term mission design and in supporting operations development by bringing out the problems that will be encountered in the operations development process.

To generate the scenario, the data acquisition simulation tool selects, within mission and instrument constraints, as many as possible of the targets requested by the investigators that it will be able to acquire in the next 16 days. In the current version of the scenario, science targets are given priority over the global mapping objective.

When the latest iteration of the 16-Day Scenario was run, operated wherever possible responding to an actual ASTER operating environment, 42% of the 1100 designated targets were acquired, 15% of the global land surface was covered with global map observations (all 3 radiometers), and 19% of the land was observed in stereo. In this iteration, science targets were not prioritized, cloud cover was not considered for the global mapping and TOOs were not included. These constraints and other s not yet considered will certainly impact the results of operating the simulation tool. They will be included in future versions of the simulation tool.

A visualization and planning tool derived from this simulator and a current EOS multi-instrument prototyping activity will be designed and prototyped. The objectives for the visualization tool include a digital global map that will illustrate the accumulation of data over time as a function of acquisition plan, the ability to scan through time and to zoom, the ability to add specific targeting and TOOs as "what ifs", and the ability to separate the data from the three telescopes.

Finally, an operations concept under discussion at this time is the creation of a U.S. Instrument

Support Terminal (IST) to be used for cooperative operations activities with the Japanese ICC for designing, developing and implementing an automatic DAR prioritizing process that would yield a conflict-free prioritized list of targets to the ICC.

4. CONCLUSIONS

The system and science operations planning and development processes require close collaboration between the U.S. and Japanese teams. There has already been a very successful meeting of the minds for everyone's benefit at the Working Group meetings and Joint Science Team meetings where surprises in the form of operating constraints were worked together to find best possible solutions. This spirit of cooperation that has successfully addressed duty cycle, for example, promises to smooth the interfaces.

This is a very exciting Project not only from the perspective of the science contributions, but from the diversity of the groups working the operations issues and the challenge of making it all come together.

ACKNOWLEDGMENTS

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

The authors wish to acknowledge Robert A. Hekl whose work on the 16-Day Scenario is reported here and both Robert Hekl and Susan E. Borutzki for their parts in the operations planning activities and their contributions to this paper.

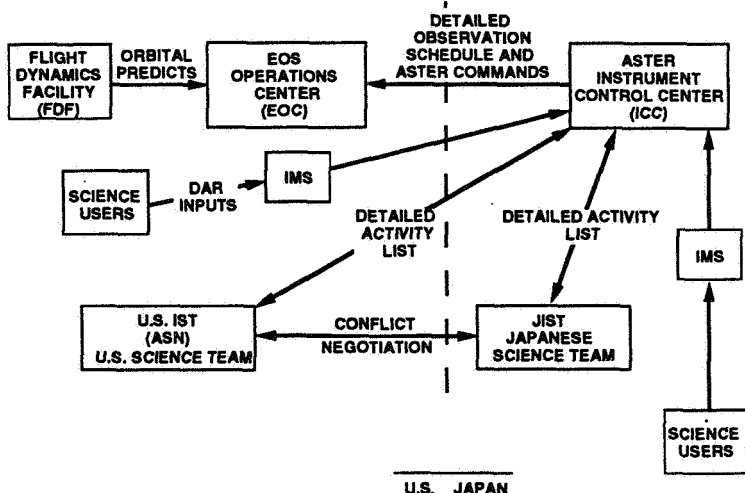


Figure 1. Flight Operations Segment (EOS)

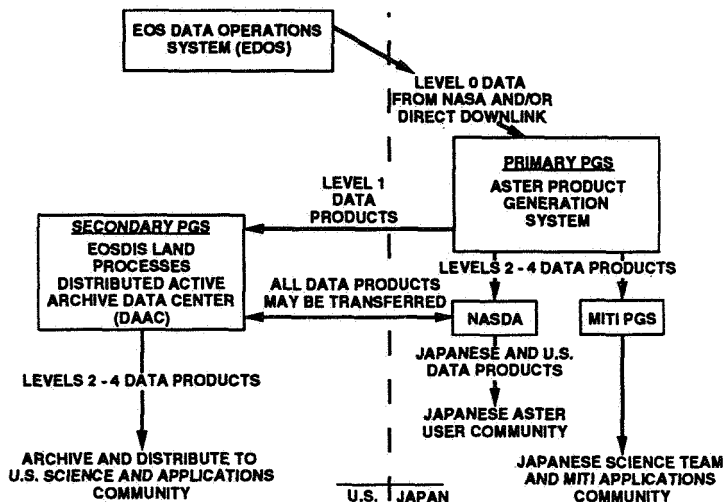


Figure 2. Science Data Processing Segment (DPS)

GROUND OPERATION OF ROBOTICS ON SPACE STATION FREEDOM

A.4

N 9 4 - 2 3 8 4 1

Z. A. (Alex) Wojcik
Manager, MSS Space Operations
Canadian Space Station Program
Office (613) 990-6660
Canadian Space Agency,
P.O. Box 7277
Vanier, Ontario
Canada K1L 8E3
Fax (613) 954-2272

David G. Hunter
Automation and Robotics
Engineering Manager
Canadian Space Station Program
Office (613) 998-6902
Canadian Space Agency,
P.O. Box 7277
Vanier, Ontario
Canada K1L 8E3
Fax (613) 954-2272

Marc R. Cantin
Robotics Operations Officer
Canadian Space Station Program
Office (613) 993-5800
KEOPS Informatique
266 Notre-Dame St. West,
Suite 201
Montréal (Québec)
Canada H2Y 1T6
Fax (514) 845-7274

ABSTRACT

This paper reflects work carried out on Ground Operated Telerobotics (GOT) in 1992 to refine further the ideas, procedures, and technologies needed to test the procedures in a high latency environment, and to integrate GOT into Space Station Freedom operations.

Space Station Freedom (SSF) will be in operation for 30 years, and will depend on robots to carry out a significant part of the assembly, maintenance and utilisation workload. Current plans call for on-orbit robotics to be operated by on-board crew members. This approach implies that on-orbit robotics operations use up considerable crew time, and that these operations cannot be carried out when SSF is un-manned.

GOT will allow robotic operations to be operated from the ground, with on-orbit crew interventions only when absolutely required.

The paper reviews how GOT would be implemented, how GOT operations would be planned and supported, and reviews GOT issues, critical success factors and benefits.

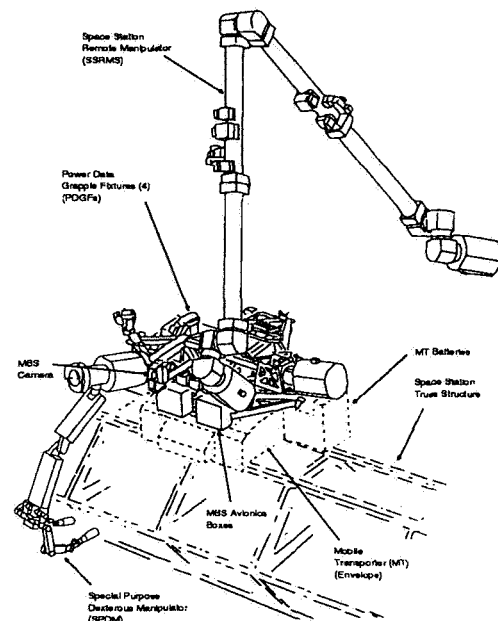
1.0 GROUND OPERATION OF ROBOTICS ON SPACE STATION FREEDOM

The International Space Station Freedom is the largest international development project ever undertaken. Canada is a partner in SSF with the United States, Europe (European Space Agency, (ESA)) and Japan (National Space Development Agency of Japan, (NASDA)). The SSF will provide a permanent base in low earth orbit, for conducting space research in areas such as astronomy, life sciences and materials research.

SSF will be launched in late 1995, and will be in operation until the year 2030.

Canada's role within the SSF Program is to design,

develop, implement and support operation of the Mobile Servicing System (MSS), an advanced on-orbit robotic system.



2.0 MSS ROLE AND OPERATION

The role of the MSS is to assist crew members in building, maintaining, utilising and in servicing SSF payloads, and to assist in the preparation for Extra Vehicular Activities (EVA) over the life of the SSF. During the SSF assembly phase, the maintenance workload is projected to be relatively high, possibly beyond the capacity of the resources currently planned to carry it out.

The current SSF Baseline Operation Plan calls for all robotic operations to be controlled by SSF crew. This approach implies that Extra Vehicular Robotics (EVR)

will only be possible when SSF is manned, and when crew resources are available, i.e. when EVR activities are in the plan. There will effectively be very little crew resource buffer available to meet any significant increase in the EVR workload, such as if the maintenance workload were to increase significantly beyond current projections.

3.0 GROUND OPERATED TELEROBOTICS

3.1 Description

Given a possible increase in the robotic workload, an approach to EVR where simple or repetitive robotics tasks could be executed without the use of crew members would appear desirable. Such an approach would imply ground-based operation of MSS robots, i.e. Ground Operated Telerobotics. When fully implemented, GOT is the end-to-end execution of MSS tasks using ground-based personnel only. This paper will continue to refer to EVR as crew operated robotics, and GOT as ground operated operation of MSS.

3.2 Operational Concept

- Under full GOT, the MSS will be operated and controlled completely from the ground. Crew, when available, would only need to be involved in robotics operations when those operations interfered with other SSF activities, or when required to carry out specific difficult operations.

GOT, as a relatively independent and unconstrained resource, would facilitate the Operations Planning of SSF, since GOT would usually be available for special tasks, workarounds, emergencies, etc.

- The GOT operator would use a stored procedure to command the MSS. This procedure would have been developed, verified and validated ahead of time by ground support personnel, and would include all steps required to execute the assigned task.
- The GOT operator would release commands step-by-step, after having verified the successful completion of the previous step, simulated the next step and verified that the simulation results were consistent with the real environment on SSF.
- After releasing a given command, the GOT operator would watch the real MSS follow the simulated image, to ensure proper execution of that step.
- The GOT workstation would be on the ground, in the NASA Space Station Control Centre (SSCC) front-room, and would not use the "joy sticks" used on SSF.

- Should a malfunction occur, the GOT operator would intervene and/or pass the command of GOT over to crew members (i.e., back to normal EVR) if they are available on board SSF.
- GOT would not take place concurrently with unrelated EVA.
- GOT would be implemented in a carefully planned and integrated step-by-step fashion. A proposed stepped approach is discussed later in this paper. Each key step of GOT implementation would be verified and validated before the eventual inclusion of GOT into the SSF Program.

3.3 Impact on Robotics Support Team

The GOT Operator would be supported by the Robotics Support Team in exactly the same fashion as when a crew member is operating MSS.

The planned training and certification program for this staff would be upgraded, in order to support GOT, which is more flexible and dynamic than baselined EVR. The resulting job description and responsibilities would improve the job quality of the ground staff, and provide an enhanced career path for this staff.

3.4 Operations and Support Systems

GOT operation would use all Baseline EVR support systems. These would include the latest versions of:

- Path Planning. Collision Avoidance and Predictive Display Systems, using an SSF World Model when it becomes available.
- Artificial Vision Function (AVF). AVF would provide essential data to assist in capturing, aligning and berthing actions, by giving GOT operator essential information on position, attitude and movement of objects.
- Fail Safe Procedures and Alert System. GOT will require that these procedures be reviewed and upgradeable.
- Data and Video Telecommunication. It is estimated that GOT would not result in a significant increase in the Data/Video transmission requirement.

3.5 Space to Ground Latency

The major operational difference between EVR and GOT would be the latency of data and video signals. In EVR, the delay between the issuing of a command by the crew at the Control Station and the command

starting its execution on MSS is virtually zero. In the case of GOT, the round-trip delay for command and data is of the order of 9 sec. Video images are delayed approximately four seconds from SSF to SSCC, and must be re-synchronised with the data.

Given this latency delay, the implementation of GOT will involve the development of specific Operations Plans and Procedures, and a stepped approach to that implementation. The safety aspect will also be addressed in the development of these procedures and in the implementation.

4.0 GOT IMPLEMENTATION PHASES

While GOT may not replace EVR for some operations, it is important to note that the Baseline Operations Plan (BOP) calls for the majority of robotic compatible tasks to be relatively simple. Based on this assumption, three areas of phasing have been defined: Control Level, Type of Activity and Point of Command.

4.1 Control Level

Control of SSF robotics would start with the baselined EVR approach, and would evolve through a gradual transfer of control to the ground operator (GOTO), using crew in a decreasingly active mode and an increasingly supervisory mode. During this phase, crew control could also be used for critical maneuvers, e.g., docking, latching, storage, etc., while simple actions such as translation would be carried out using GOT.

This evolution would culminate with full control of nominal SSF robotics by ground operators, ie. full GOT, with crew attention only in emergencies and when available.

4.2 Type of Robotic Activity

The nature of the activities to be carried out using GOT will evolve as skills, quality of procedures and knowledge grow. GOT will initially be used for simple translation of MSS, with or without cargo, including start-up, health status, return-to-base, safeing. Cargo may be any ORU, OMI or other on-orbit item. GOT would then evolve to contact (docking, grasp/de-grasp) and visual inspection tasks. The final step would be end-to-end control of a task, such as an ORU replacement.

4.3 Point of Command on the Ground

Another step in the implementation of GOT can be the actual location of the ground staff controlling GOT. Current GOT planning calls for the NASA Space Station Control Centre (SSCC) to be the ground command

location. CSA will ensure that the implementation of the Canadian Engineering Support Centre (CESC) will not preclude the eventual control of GOT from the CESC, with the NASA Control Center in supervisory and support mode. Such a long-term evolution would only take place after GOT has been well tested and proven, with the full and active support of NASA Mission Operations Directorate (MOD) at JSC.

5.0 SPDM GROUND CONTROL STUDY AND TEST

In September of 1992, the Robotics Systems Evaluation Laboratory, located at the Lyndon B. Johnson Space Center in Houston, Texas, conducted a Special Purpose Dexterous Manipulator (SPDM) Ground Control Study. This study was conducted to evaluate the feasibility and impact of implementing ground control for robotic operations on Space Station Freedom.

This test attempted to answer some of these questions, by performing a representative robotic task using four different modes of operation: teleoperation without and with ground based time delay, teleoperation with ground based time delay and a predictive display, and a semi-autonomous/supervisory mode with ground based time delay. Teleoperation without time delay effectively replicates EVR, and provides a basis for comparison between manual EVR operation of MSS and manual operation of the robot from the ground, with the time delay.

The usefulness of predictive displays was evaluated by comparing the second and third control modes.

The study also compared the performance of teleoperation with semi-autonomous modes for robotic ground control. In this study, semi-autonomous mode refers to the use of a combination of teleoperation and autosequence commands to perform the task.

The complete SPDM Ground Control Study will eventually be made up of several test phases. The first phase of testing involved performing a combined inspection and EVA worksite setup task, and an ORU acquisition and stowage, using a telemetry roundtrip delay of 9 seconds and a command/video roundtrip delay of 6 seconds.

Preliminary results have shown that the combined use of a predictive display system and stored procedures reduces both the number of errors from the operators and the elapsed time required to perform the tasks. The resulting elapsed time is of the same order of magnitude as basic teleoperation, without time delays, ie. normal EVR.

6.0 GOT IMPLEMENTATION: OTHER ISSUES

6.1 MSS Command Hardware and Software

In order to execute MSS commands issued from the ground, the SSF Control software must be modified, as the current design requires that the MSS operator "arm" the system by physically depressing a switch on one of the joysticks, before "firing" a stored command or moving the stick.

6.2 Impact on SSF

It is likely that using GOT will increase the actual usage of MSS robots, and increase propellant and power consumption aboard the MSS. The resulting wear-and-tear on all components must be monitored, with maintenance and evolution of the design being followed up carefully. Spares, logistics, re-supply and robotic compatibility of spares are further design and planning issues which must be addressed.

6.3 Verification of GOT

Each GOT implementation step must be verified and validated (V&V) by ground personnel and by crew. This V&V process must be built into the MSS Mission plans, as soon as enough MSS hardware and software are available to begin executing GOT, i.e. MB-7. Another possible method of validation would be by emulating GOT using the SS Orbiter and its robotic arm, the SRMS.

7.0 BENEFITS OF GOT

GOT benefits have been identified and quantified to varying degrees in several papers and documents.

7.1 Time-Delayed Remote Operation and Maintenance of Space Station Freedom

This IAF 1991 paper analyzes a particular MSS ORU replacement operation, similar to the SPDM Ground Control Study and Test discussed elsewhere in this paper. The conclusions of this paper were consistent with the SPDM study, even though this paper was of a more theoretical nature, and was used as a starting point for the GOT concept.

7.2 SPDM Robot/Astronaut Comparison with Respect to SSF Operation

This IAF 1991 paper compared end-to-end elapsed time, (i.e., crew task time) to change a given ORU using robotics in one case and Extra Vehicular Activity (EVA) in the other.

The conclusions of this paper were also consistent with the SPDM study, in addition to highlighting the reduction in crew attention which can be achieved by using GOT.

7.3 Other benefits of GOT

GOT in itself is a virtually limitless resource, bounded only by the reliability of the MSS, and the availability of SSF resources and MSS spares. In addition to providing direct savings in IVA and EVA time, GOT will benefit the SSF program in several other ways:

- **Availability of Robotics.** In the short and the long term, GOT would allow space robotics to become an unlimited resource, capable of executing not only maintenance tasks (corrective, preventive) but also janitorial tasks, inspections, calibration of equipment and the execution of some space experiments or other dangerous tasks.
- **Greater Effectiveness and Availability of SSF.** Using GOT to reduce the maintenance backlog will result in an improved and more efficient SSF, and would reduce the impact on SSF of an accumulation of individually non-critical ORU/OMI failures. Should the pre-MTC Mission Build flight schedule slip and leave SSF unmanned for even longer periods, GOT will continue to ensure Maintenance Actions are carried out.
- **Greater Flexibility in Short Term Planning.** Using GOT to address urgent or time consuming tasks, without crew availability restrictions, would give mission planners more freedom to plan for productive work for the crew, and more freedom to react to emergencies during both manned and un-manned periods.
- **GOT Expert Systems.** Over time, GOT would allow the construction of a knowledge base from which better Procedures, Mission Planning and Execution Tools can be developed. Such databases and associated Expert Systems will not only improve the operation of SSF but will also have direct commercial applications.
- **Enhanced Role of Ground Staff.** Implementation of GOT would increase the importance and influence of Robotics related ground positions. This would be beneficial in terms of morale, career path and personal development, and result in an increased role for trainers and related ground installations.
- **Cost Effectiveness.** In all the above cases, GOT would also reduce the cost of the overall program, since ground personnel and equipment is almost certainly cheaper to build and use, and more flexible

than on-orbit personnel and equipment.

- GOT in the Commercial World. The development and implementation of GOT has direct commercial applicability. Space represents extremes in distance, latency, hostility of environment and complexity.

8.0 RISK ASSESSMENT

On-Orbit safety is of vital importance. GOT calls for the implementation of new techniques and procedures, all of which have a direct impact on the day-to-day operation of SSF. Since GOT involves the movement of important masses, this movement will imply a constantly changing "World" in SSF, and since GOT will release crew resources to carry out other activities, it is critical that all GOT related Principles of Operation, Procedures, and actual operations be thoroughly developed, documented, tested, validated, approved and integrated into the SSF operations plan.

R&D requirements are well in hand, since GOT requires no extra techniques or support systems above and beyond those already planned for SSF.

From the programmatic point of view, GOT must be fully integrated into the document flow, starting at the Operations Concept and Plans level, and flowing down to requirements, specifications, design and implementation.

9.0 CRITICAL SUCCESS FACTORS

To be successfully implemented and used, GOT needs the following factors to be addressed.

Implementation of GOT will require funding over and above current SSF program budgets. It is also vital that SSF and MSS not be re-scoped in such a way as to preclude the implementation of GOT. Further, for GOT to be successfully integrated into SSF operations, the concept must be fully supported by all NASA contractors and partners.

Finally, several elements of MSS, both on-orbit and on the ground, will need to be modified or improved in order to implement GOT.

- MSS hardware will need to be monitored closely to ensure that the increased work load does not negatively impact the ability of MSS to execute mission critical tasks. The resulting spares and logistics issues must also be addressed.
- MSS software will need to be modified to allow for the execution of MSS commands which are issued from the ground.

- Management, support and staffing on the ground must allow for real-time support of GOT and for the development and validation of MSS procedures in real-time, without the involvement of on-orbit crew.
- All robotics activity on SSF, both in EVR and in GOT modes, will depend entirely on video cameras, lighting and a sophisticated Advanced Vision Facility (AVF), thereby further increasing the importance of that aspect of robotics operations on SSF.
- The SSF/MSS high fidelity simulator used in the MSS Procedure development process and during GOT execution of these procedures must be capable of operating in real-time mode, be easily updated to reflect the actual SSF "world" at any given time, and must be available to execute nominal procedures as GOT is in execution, while real-time simulating an amended procedure as the procedure in execution is being revised.

ACKNOWLEDGMENTS

Bernard G. Baker, Canadian Space Agency, Ottawa, Canada
Bryan Erb, Canadian Space Agency, Houston, Texas
Elaine Greenberg, Techknowledge, Ottawa, Canada
Norman Lee, Spar Aerospace, Houston, Texas
Denise Wagner, NASA Mission Operations Directorate, Houston, Texas

REFERENCES

1. D.L. Brown, L.R. Stevens, "SPDM Robot/Astronaut Comparison with Respect to SSF Operations", 42nd International Astronautical Congress, Montréal, Québec, October 1991.
2. D.G. Hunter, Z.A. Wojcik, D.G. Cooke, "Time-Delayed Remote Operation and Maintenance of SSF", 42nd International Astronautical Congress, Montréal, Québec, October 1991.
3. Z.A. Wojcik, M.R. Cantin, "Ground Controlled Telerobotics", White Paper, Canadian Space Agency, August 1992.
4. Z.A. Wojcik, "Ground Operation of the Mobile Servicing System on Space Station Freedom", SPIE's OE/TECHNOLOGY '92, November 1992.

MISSION PLANNING FOR SHUTTLE IMAGING RADAR - C (SIR-C) WITH A REAL-TIME INTERACTIVE PLANNING SOFTWARE*

N94-23842

Su K. Potts**

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA

ABSTRACT

The Shuttle Imaging Radar - C (SIR-C) mission will operate from the payload bay of the space shuttle for 8 days, gathering Synthetic Aperture Radar (SAR) data over specific sites on the Earth. The short duration of the mission and the requirement for real-time planning offer challenges in mission planning and in the design of the Planning and Analysis Subsystem (PAS). The PAS generates shuttle ephemerides and mission planning data and provides an interactive real-time tool for quick mission replanning. It offers a multi-user and multi-processing environment, and it is able to keep multiple versions of the mission timeline data while maintaining data integrity and security. Its flexible design allows one software to provide different menu options based on the user's operational function, and makes it easy to tailor the software for other Earth orbiting missions.

Key Words: mission planning, software, timeline, space shuttle, graphics, operations

1. INTRODUCTION

Shuttle Imaging Radar - C (SIR-C), together with the X-Band Synthetic Aperture Radar (X-SAR) of Germany, is a mission currently scheduled for three flights on the space shuttle, the first of which will be in October 1993. The second and third flights are scheduled in two different seasons of 1995 and 1996.

SIR-C's instrument is a Synthetic Aperture Radar (SAR). It will operate from the shuttle's payload bay, gathering 50 hours of data over numerous sites, and it will return with the shuttle after each flight.

* The work described in this paper was performed at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

** Technical Group Leader
Launch and Earth Orbit Mission Analysis Group
Mission Design Section, JPL
Member AIAA

The data returned from this mission will provide the science community with simultaneous multi-frequency, multi-polarization, and multi-parameter radar imagery from space. A complete science plan is documented in the SIR-C Science Plan (Ref. 1).

Each flight of SIR-C will last 8 days. Mission operations will be performed from a Payload Operations Control Center (POCC) at the Johnson Space Center (JSC). Due to the short duration of the mission, 24-hour continuous operations is planned. Timely replanning and quick responses to real-time anomalies are essential to the success of the mission.

The Planning and Analysis Subsystem (PAS) is a part of the SIR-C Mission Operations System (MOS) currently in development at the Jet Propulsion Laboratory. This paper will describe the SIR-C mission planning scenario and the design of PAS which supports that scenario with a multi-user, multi-processing environment using interactive software which allows real-time replanning.

2. MISSION PLANNING SCENARIO

Pre-flight mission planning activities are currently under way to develop a baseline mission timeline. Datatakes are a major part of the baseline timeline, which also includes playbacks of recorded data and live downlinks. During the mission, these datatakes will be replanned as necessary based on the navigation data received from JSC.

Major perturbations to the orbit, such as shuttle attitude maneuvers required for the mission and trim burns which may be required to maintain the mission trajectory, contribute to relatively large uncertainties in orbit determination by JSC Navigation. The errors in atmospheric modeling at SIR-C's low average reference altitude of 215 km further contribute to the uncertainties in orbit prediction. These errors quickly propagate to greater uncertainties as a function of time. Preliminary analysis has indicated that after 6 hours of orbit propagation, the science requirement of imaging a site within ± 4 km 90% of the time will not be satisfied.

In order to meet the site imaging accuracy requirement, SIR-C mission replanning will be performed in two stages: long-term planning in 12-hr cycles to update the planned datatakes in the baseline timeline, and real-time planning to update the radar parameters for each datatake scheduled during the long-term planning cycle.

Every 12 hours, an ephemeris will be generated with the most current navigation data from JSC. This ephemeris data, which will span from the beginning of the planning cycle to the end of the mission, will be compared against the current baseline ephemeris. The extent of the differences found from this ephemeris comparison, the status of data collection from the start of the mission, and any new mission constraints will be incorporated into the long-term replanning of the datatakes scheduled for the next 12 hours. Some planned datatakes may be deleted, some new ones may be added, and some may be updated with parameter changes.

The portion of the mission timeline updated during the long-term planning cycle is then verified and updated again during real-time planning. With the navigation data received from JSC on the average of once per orbit, approximately every 90 minutes, ephemeris data for the next few hours are generated and used to update the radar setup parameters for each upcoming datatake. Deletions and additions of datatakes are not a part of nominal real-time planning activities, although the real-time planner must be able to perform such activities if it becomes necessary in the event of an anomaly.

The updates to the timeline planned during the long-term planning cycle are approved by the appropriate persons of the SIR-C operations team as well as by JSC. Once signed off, they are incorporated into the approved timeline, which is used by the entire operations team to carry out their responsibilities.

3. FUNCTIONAL REQUIREMENTS AND DESIGN CHALLENGES

The development of the Planning and Analysis Subsystem (PAS) was presented with some interesting design challenges. While the PAS must provide the tools necessary to perform the functions described in the planning scenario above, it must also satisfy the needs of the entire operations team, who must be able to view the approved mission timeline at all times.

To support the mission planning activities, the PAS must perform high precision calculations for generation of mission planning data, must present the data to the users such that right decisions and tradeoffs

can be made efficiently, and must provide a quick response time for real-time replanning.

Some of the major functions of the PAS include processing of the state vectors from JSC, generation of ephemeris data and mission planning data, providing tools to develop a timeline of events, graphical displays of ground tracks, sites, and radar swaths against a world map, calculation of radar parameters, and constraint checking.

It must be able to maintain the approved timeline while the changes to the datatakes are planned, approved, and finally incorporated into the approved version. And since there are two planners, a long-term planner and a real-time planner, who are both making updates to the approved timeline, the PAS must be able to keep their activities separate from each other, and furthermore, keep their activities separate from the approved timeline data until the changes are ready to be incorporated.

There are several users of the PAS who must have the write access to specific mission data in order to support mission replanning and operations activities. All other users must be prevented from writing to the mission data. It is crucial that the write access to the data is strictly enforced.

The mission planning data, as well as the approved mission timeline data, must be viewable by each position on the operations team, and the set of data viewed by one position can be different from that viewed by another depending on the functions of the positions. It is also possible, during the mission, that an anomaly may result in a change in one's operational position, requiring that person to have access to the tools necessary to perform the functions of the new position. The provision of the appropriate software tools must be accomplished very quickly to minimize interruptions to the mission planning process.

The goal was to develop one PAS system that provides the necessary control of the software and the data to guarantee its security and the integrity, has the flexibility required to support a very dynamic mission planning and operations scenario, and has the adaptability for responding to anomalies.

4. SYSTEM ARCHITECTURE AND DESIGN

The PAS system consists of a computer network, a central database, and software tools to aid in mission planning. Described below are some of the major features of the PAS system which satisfy the functional requirements and provide answers to the design challenges.

4.1 Computer System and Data Storage

The PAS operates in a local area network which includes one host computer and several workstations which communicate with the host via ethernet.

One operational copy of the PAS software and the data reside on the host, and they are accessible by all workstations in the network. This configuration allows all users to have access to the most current data as they are continually updated. It also guarantees that all users are using the same version of the software

Some of the data in the host computer are stored as ASCII files, but most of the data are housed in a relational database. The database environment and the PAS software together control the multiple user data access in order to ensure data integrity and security.

The PAS fully utilizes the multiple windowing capabilities of the workstations as well as colors and controls using the mouse. These features increase user productivity, which is an important consideration for quick real-time replanning.

4.2 Central Database and Data Access Control

The PAS has a central database which is located on the host computer, and it holds most of the mission planning data and the timeline data. The data are organized into many tables with specific links defined in order to provide faster response time for more time consuming activities.

All users of the PAS are assigned to one of several groups, and each group has different access privileges to the database. The read access to these tables is granted to all groups, but the write access to each table in the database is limited to those groups

responsible for maintaining the data in that table. In most cases, the write access to a table is limited to only one group.

The database also serves as the main interface between the software modules. The input data to each module are read from the database, and the output data are written back to the database, either to be read by another module or to be displayed to the user.

4.3 One PAS for Multiple Users

The software modules are integrated into one software package with a top level menu that reads the user id, retrieves the group to which the user is assigned, and presents only those menu options that are granted for that group. This design of the PAS allows one software to be used by several users performing different functions.

If a user is reassigned to perform the functions of a different group, one entry in one table is modified to assign the user to the new group. In less than a minute, this user will have a new set of menu options allowing him/her to have the tools necessary to perform the new functions.

Figure 1 shows a small portion of the PAS top level menu and illustrates the differences between the menu options displayed for two different groups.

The menus are designed to support multi processing, with the number of processes and the types of processes selected by the user. Two users in the same group, although they have access to the same menu options, can choose to initiate only those processes applicable to their operational function. With this flexible design, the PAS display can have several forms, each tailored to the specific needs of each user. An example is shown in Figure 2.

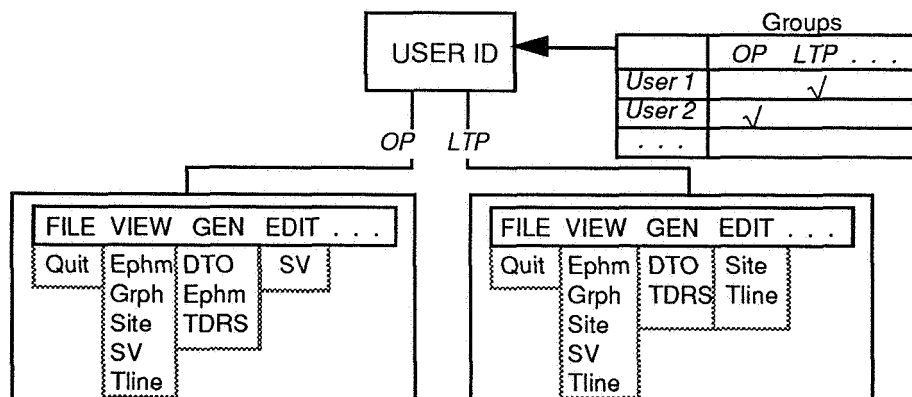


Figure 1. PAS Main Menu Example

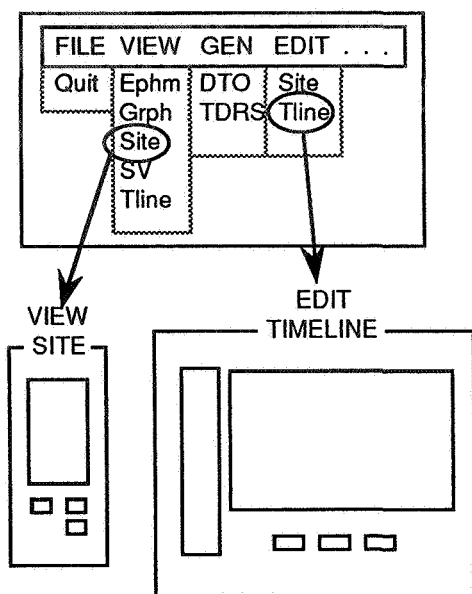


Figure 2. An Example of User Defined Multi-processing

4.4 Timeline Development Tools

The timeline development process begins with an arrival of the navigation data from JSC and ends with an approved timeline from which the commands are generated.

After the navigation data are electronically received from JSC, a coordinate transformation is performed to make it compatible with the coordinate system used by the PAS, and then it is stored in the database. User-selected state vectors from the database are read into an orbit propagator which generates the ephemeris data as well as the ground track data and stores them in an ASCII file. Certain parameters from the ephemeris file are ported into the database so that they can be manipulated easily.

Sites selected by the SIR-C Principal Investigators are stored in the database also. These data can be updated interactively using the site editing module.

The site data and the ephemeris data are used by another module which determines the datatake opportunities by calculating the geometry necessary to image the sites. These datatake opportunities, which are stored in the database, can be viewed, sorted, searched, and finally selected as datatakes.

Once an opportunity is selected as a planned datatake, the links between the different types of data are utilized to present complete information associated

with a datatake. For example, the link between the opportunity data and the site data is used to retrieve the requested radar setup parameters for that site. The module which calculates the radar setup parameters uses the requested values as an input to determine the optimal radar setting for that datatake.

Since all data will be recorded onto tapes on board the shuttle, sophisticated tape management modules are available to automatically assign datatakes to tapes and plan tape loading and unloading events. The automatic tape assignments can be modified using the manual mode. Playbacks of data recorded on tapes are easily planned by displaying the content of the tapes and selecting the datatakes to be played back. Useful tape information, such as the percentage of each tape used, is readily available to the user.

Figure 3 shows the data storage, the external interfaces, the processes, and the interfaces between the processes described in this section and in sections 4.5 to 4.8. Note that this diagram illustrates only the functions discussed in this paper; it does not represent the entire PAS system.

4.5 Constraint Checking

Constraint checking, which is a very important part of mission planning, is also supported by the PAS. It is used to resolve conflicting activities, to check against power and energy constraints, and to verify tape assignments. The availability of Tracking and Data Relay Satellites (TDRSs) is also checked when data playbacks or live downlinks are planned.

Any violations to the mission rules are flagged by the software, and the generation of commands will not be initiated until the timeline either passes the constraint checks or the flags are acknowledged and overridden by the user.

4.6 Graphics Displays

The PAS has color graphics displays which provide the user with visual representations of the mission.

The mission planning data, such as ground tracks, sites, and potential radar swaths can be displayed graphically against a map of the world. Several options are available to tailor the display to the needs of the user. These options include zooming into a small area on the map, selection of overlays, labeling of orbit numbers and times, and output to a printer and a color plotter. One of the options most useful for the SIR-C mission is the capability to plot datatakes directly from the timeline, which will be used to plot daily coverage maps before, during, and after the mission.

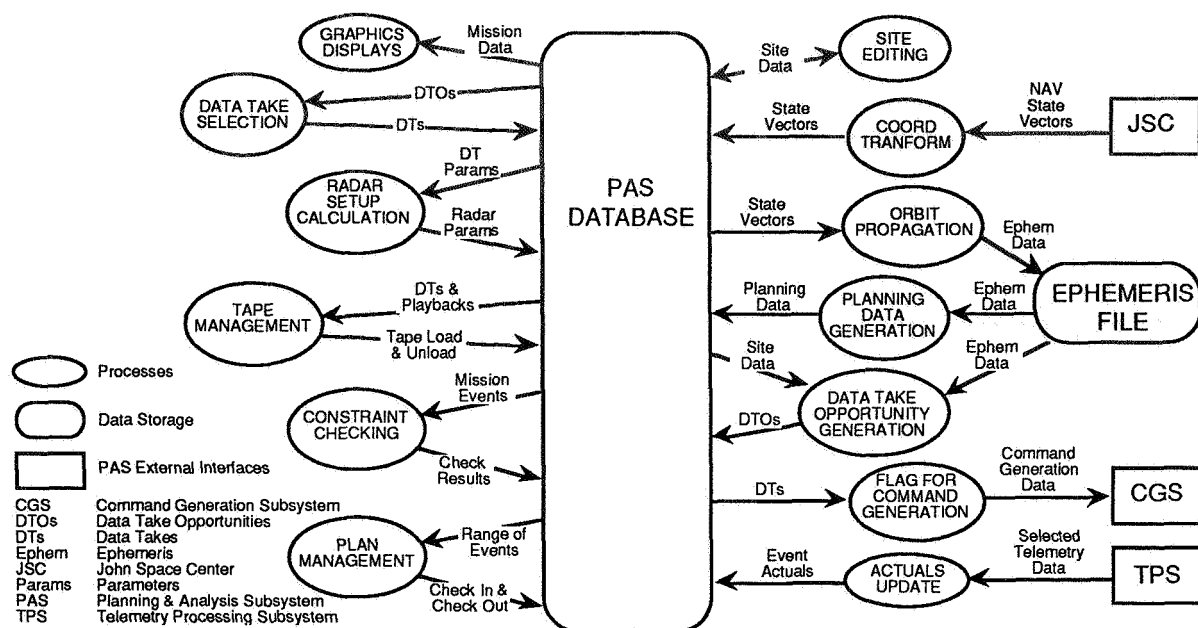


Figure 3. PAS Mission Planning Functional Diagram

All mission events stored in the database, both executed and planned, can be displayed on a graphics timeline. Colors are utilized to distinguish events and to alarm constraint violations. To assist users in planning certain activities, also shown on the same display is information such as TDRS availability and whether the shuttle is in daylight or night.

Color graphics is also used to display the power used for each datatake and the energy consumption as a function of time. This display provides an overview of the power and energy status, therefore allowing the user to make appropriate decisions if the power and energy constraints are violated.

4.7 Working Plan vs. Approved Plan

While the approved plan in the database is used by the operations team, the planners can work on updates to the approved plan in separate working plans. Since there are two mission planners, a long-term planner and a real-time planner, there will be at least two working plans.

When a portion of the approved timeline is identified for updates, that portion is checked out by a planner, and the software marks those events in the approved timeline so that the operations team can see which portion is in the process of being updated by a planner. The software prevents the other planner from checking out the portion of the timeline already checked out by the first planner. Operational procedures dictate that the two planners do not work

on the same portion of the approved timeline; the software safeguards against accidental overlaps.

4.8 Updates with Actuals

The Telemetry Processing Subsystem of the SIR-C MOS receives and processes the telemetry data. Some parameters in the telemetry data are routed to the PAS, and they are used to update the events in the approved timeline with the actuals. For example, the datatakes are updated with the actual on and off times as well as the actual tape start and stop identification numbers. These updates are performed immediately following a completion of an event.

Actual energy consumption data are received from JSC every 12 hours, and this information will also be entered into the database and compared against the predicted values. Adjustments in the software parameters can be made if there is a significant difference between the predicted and the actual energy consumption so that the remainder of the mission can be planned with more accurate energy constraints. The adjustments are made by simply updating the software input parameters stored in the database; no changes to the code will be necessary.

Figure 4 is a graphical representation of different stages associated with the timeline. Events are initially planned in the pre-flight baseline timeline; they are then updated during a long-term planning cycle; further updates are made during real-time replanning; and after the execution of the events, they

are updated with actuals. At the end of the mission, the timeline becomes the as-flown timeline.

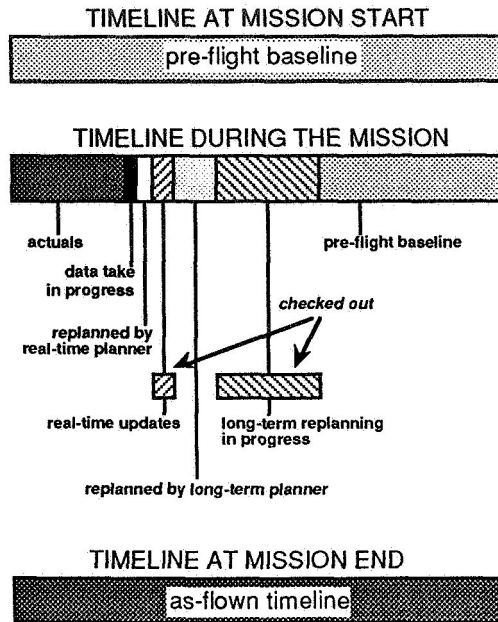


Figure 4 Stages of Mission Timeline

4.9 Parameter Updates in Response to Anomalies

Similar to the way an adjustment is made to the energy consumption software module, other software parameters can be changed to respond to certain anomalies during the mission. Many parameters relating to mission planning and operations are stored in tables in the database, making it possible for quick updates with very little impact to the mission planning process.

For example, there are 16 different Pulse Repetition Frequencies (PRFs) available for SIR-C datatakes. Nominally, all 16 PRFs are considered when calculating radar setup parameters for each datatake. If one PRF becomes unusable as a result of an anomaly, that PRF will be marked as 'not available' in the database, and it will not be presented to the user in the list of possible PRFs.

5. SOFTWARE DEVELOPMENT

The PAS software is currently being developed in a VAX/VMS environment with DecWindows/Motif providing the multi-windowing capability. The database was developed with the Ingres Relational Database Management System, and Ingres Windows4gl is being used to write a majority of the software modules of the PAS. The graphics displays

of maps and overlays utilize the VAX Graphical Kernel System (GKS). Fortran is the language used for the 3gl programs.

Two sets of software have been inherited from the Mission Design Section of JPL for the development of PAS: the Vector Library of the Multimission Analysis Software Library (MASL), and the Planetary Observer High Precision Orbit Propagator (POHOP) of the Planetary Observer Planning Software (POPS) (Ref. 2)

The PAS software is developed in small modules. Each module is integrated into one of several applications, and one main application, which provides the main menu, calls other applications as they are selected by the user. This development strategy offers flexibility in software development, simplifies the debugging processes, provides easier software maintenance and configuration control, and accommodates future upgrades and changes.

6. APPLICATION TO OTHER MISSIONS

Although PAS is being developed specifically for the SIR-C mission, many features of the system can be used for other Earth orbiting missions. Some capabilities of the current working version of the PAS have been used to perform preliminary mission design for EosSAR and for other Earth orbiting mission proposals.

Because PAS is flexible and modular in its design, changes necessary to support other missions can be performed fairly easily. The preliminary versions of the SIR-C PAS have been delivered to the X-SAR project, who plans to make modifications and additions to support X-SAR mission planning. SIR-C and X-SAR mission planning will be performed jointly from a Payload Operations Control Center at JSC.

7. ACKNOWLEDGMENT

I thank Daren Casey, Richard Lee, and Gina Walton at JPL, and Mark Chapoton at Midcom Corp., who have contributed to the design of the PAS and who are currently involved with the development effort.

8. REFERENCES

1. Shuttle Imaging Radar - C Science Plan. JPL Publication 86-29. 1 September 1986
2. Smith Jr., John C. The Planetary Observer Planning Software (POPS). JPL EM 312/91-162 (Internal Document). 11 February 1991.

N94-23843

SAMPEX PAYLOAD OPERATION CONTROL CENTER IMPLEMENTATION

Daniel Mandl, Jack Koslosky, Ron Mahmot, Michael Rackley

NASA/GSFC Code 511

Greenbelt, MD 20771

Dr. Jack Lauderdale

Computer Science Corporation

1100 West St. Laurel, MD 20707

ABSTRACT

The Solar Anomalous and Magnetospheric Explorer (SAMPEX) satellite was launched in July 1992. It was the first in the NASA Small Explorer (SMEX) series. In building the real-time control center facility, several new mission support challenges had to be met; (a) CCSDS telemetry and command format, (b) 900 Kbps telemetry data, and (c) shorter turn-around time for control center development than previous missions

The SAMPEX Payload Operations Control Center (POCC) was also the first control center for a new satellite to be based on the Transportable Payload Operations Control Center (TPOCC) system architecture and methodology. This approach has both guided the implementation of the SAMPEX control center and provided some of the building blocks. By using the TPOCC architecture to build the SAMPEX POCC, the real-time operations area was miniturized into one room, whereas previous missions needed multiple large rooms. The development cost of the SAMPEX POCC was reduced from previous missions and will provide for further cost savings in the future SMEX satellites.

This paper describes the system as built and some of the enhancements in progress to create this teleoperations environment.

Key Words: Spacecraft control center, Payload Operations Control Center, Teleoperations

1. INTRODUCTION:

The Control Center Systems Branch (CCSB) of the Mission Operations Division (MOD) at the NASA Goddard Space Flight Center was tasked with the creation of the Payload Operation Control Center (POCC) for Solar Anomalous Magnetospheric Particle Explorer (SAMPEX). The CCSB has created POCCs for such satellites as the Cosmic Background Explorer, the Gamma Ray Observatory, the Upper Atmosphere Research Satellite and the Extreme Ultraviolet

Explorer. Presently POCCs are being developed for missions such as the Interplanetary Physics Laboratory (WIND), the Polar Plasma Laboratory (POLAR), the Solar Oscillator Heliospheric Observatory (SOHO), the X-Ray Timing Explorer (XTE), and the Tropical Rainfall Measuring Mission (TRMM).

SAMPEX is the first in a series of NASA Small Explorer (SMEX) satellites. The SMEX satellites were intended to be low cost and to be launched 3 years from conception. The SAMPEX mission is a relatively simple mission whose purpose is to measure the elemental and isotopic composition of ions emitted from the sun with energies over the range from one to several hundred megaelectron volts. However, the fact that the mission was simple did not translate into a simple ground system. Although the SMEX series of satellites were specified as low cost missions, with the low cost being achieved via the use of less redundancy and higher risk operations, the higher risk introduced into the space segment introduced additional requirements into the ground system in order to assure the health and safety of SAMPEX. Also, SAMPEX was the first satellite to introduce the use of the Small Explorer Data System (SEDS) as the on board computer. The SEDS is a 80386-CPU based computer and features dynamic memory management. The SEDS, although allowing for greatly enhanced capabilities for the Flight Operations Team (FOT), introduced an additional level of complexity in terms of managing the flight software, tables and the stored command processors. The SEDS features a 32 Mbyte solid state recorder which permits recorded telemetry to be dumped data in forward order and for data to be stored and dumped simultaneously.

2. CHALLENGES:

There were a number of compelling challenges in the creation of the SAMPEX POCC. The greatest

challenge was the relatively short development cycle. The launch date was driven by both a need to launch as close as possible to the solar maximum and the desire by NASA to demonstrate that low cost, quick turnaround missions were feasible. In light of ongoing budget cuts, streamlining of mission costs has become a primary driver for NASA's projects. In the past, launch has typically occurred more than 36 months after the POCC System Requirements Review (SRR). For SAMPEX, launch occurred in only 20 months after the POCC SRR. This consolidation of the schedule occurred only through a dedicated team effort in which many traditional development and testing methods had to be modified to increase efficiency.

Another challenge was the use of the CCSDS protocol, which is a packet based approach for both command uplinks and telemetry downlinks. The challenge with CCSDS was that there was no operationally proven equipment or software upon which to base the creation of the SAMPEX POCC system.

A third challenge was that the telemetry data rate to be processed by the POCC and other elements of the ground system was 900 Kbps. This was a much higher rate than the MOD had previously dealt with. New block and frame synchronization equipment had to be developed. The engineering real-time data rate was only 16 kbps, but was enveloped in the higher data rate composite stream primarily comprised of science data.

Finally, SAMPEX was the first new mission created using the Transportable Payload Operations Control Center (TPOCC) core software and hardware components. The use of the TPOCC architecture presented challenges in coordinating the TPOCC and SAMPEX mission specific efforts and in educating the Flight Operations Team (FOT) personnel about a new operations approach.

3. SAMPEX POCC FUNCTIONAL REQUIREMENTS:

The high level functional requirements of the SAMPEX POCC were as follows:

- (1) Provide reception and processing capabilities for telemetry data
- (2) Provide the capability to playback satellite recorder data and process the recorder

engineering data portion similar to the way that realtime engineering data is processed following pass completion.

- (3) Provide real-time commanding
- (4) Provide displays and reports to monitor telemetry processing and commanding.
- (5) Provide command panel user interface
- (6) Provide a high level language (System Test and Operations Language) to control all POCC functions.
- (7) Accept project provided database of telemetry points and commands.
- (8) Provide equipment and facilities for the POCC.

4. MANAGEMENT APPROACH:

The management approach for developing the SAMPEX POCC was different from previous POCCs. The SAMPEX POCC is based on the TPOCC architecture and, therefore, also inherited its management approach. In particular, custom software and hardware solutions are avoided as much as possible. By maximizing the use of Custom Off The Shelf (COTS) hardware and software, the CCSB hopes to harness the synergy derived when industry focuses on standards.

The compliance with these standards allows for many of the POCC features to be provided as COTS components. Those features which cannot be purchased from industry are developed in-house and manufactured as reusable components, whether hardware or software. The TPOCC group acts as a focal point for system baselining of the core functionality for SAMPEX and other TPOCC-based POCC development efforts.

All of this translates into maximizing both reuse and flexibility. Over many years of POCC development, we have learned that requirements for POCCs tend to develop as operational requirements mature. Requirements can rarely be cast in concrete and therefore flexibility is a must. There are always changes requested and the quicker these changes can be accommodated, the lower the ultimate cost of development. The initial effort of creating a very flexible system tends to be higher than tailoring the system, however, in the long run there is greater system flexibility and cost savings by following this approach.

This flexibility was evident in the SAMPEX POCC

development effort. POCC software was being developed for the SAMPEX and WIND/POLAR missions simultaneously with the implementation of the core TPOCC software. In order to ensure that the functionality needed for SAMPEX would be available and meet mission requirements on schedule, implementation of essential TPOCC generic software was coordinated and shared among the TPOCC and TPOCC-based mission groups. Second, the core telemetry processing capabilities were prototyped by the SAMPEX POCC group during its design phase to ensure that the high telemetry data rate could be supported by the proposed hardware/software configuration. The coordination of implementation and early prototyping was critical in meeting the short POCC system development schedule.

Following industry standards not only allows COTS software to be incorporated into TPOCC, but also provides a path for incorporating software from other sources. For example, when groups external to TPOCC come up with good ideas or useful software, the TPOCC group will integrate these ideas or software into the core set of functionality so that all TPOCC based POCCs have access. An example of this is the ongoing effort to integrate the Generic Spacecraft Analyst Assistant (GenSAA), which is an expert system package which allows the operator to generate data driven diagrams and rule based processes.

But the real power in the management approach is the empowerment of the members of the TPOCC development team via the formation of a TPOCC working group. The TPOCC working group includes all users of TPOCC and the TPOCC group itself, contractor and government, to make decisions as to how the system will work. Team effort is emphasized with the main goal being to improve the TPOCC system in line with evolving POCC requirements from the various missions. Emphasis is placed on doing it right the first time rather than depending on external quality control.

5. DESIGN APPROACH:

The design approach taken for the SAMPEX POCC was to provide functional strings of equipment consisting of workstations and X-terminals interconnected with a VME-based Front End Processor (FEP) via ethernet. Each string provides full independent POCC functionality (see fig 1). A prime and backup string are provided in the MOR,

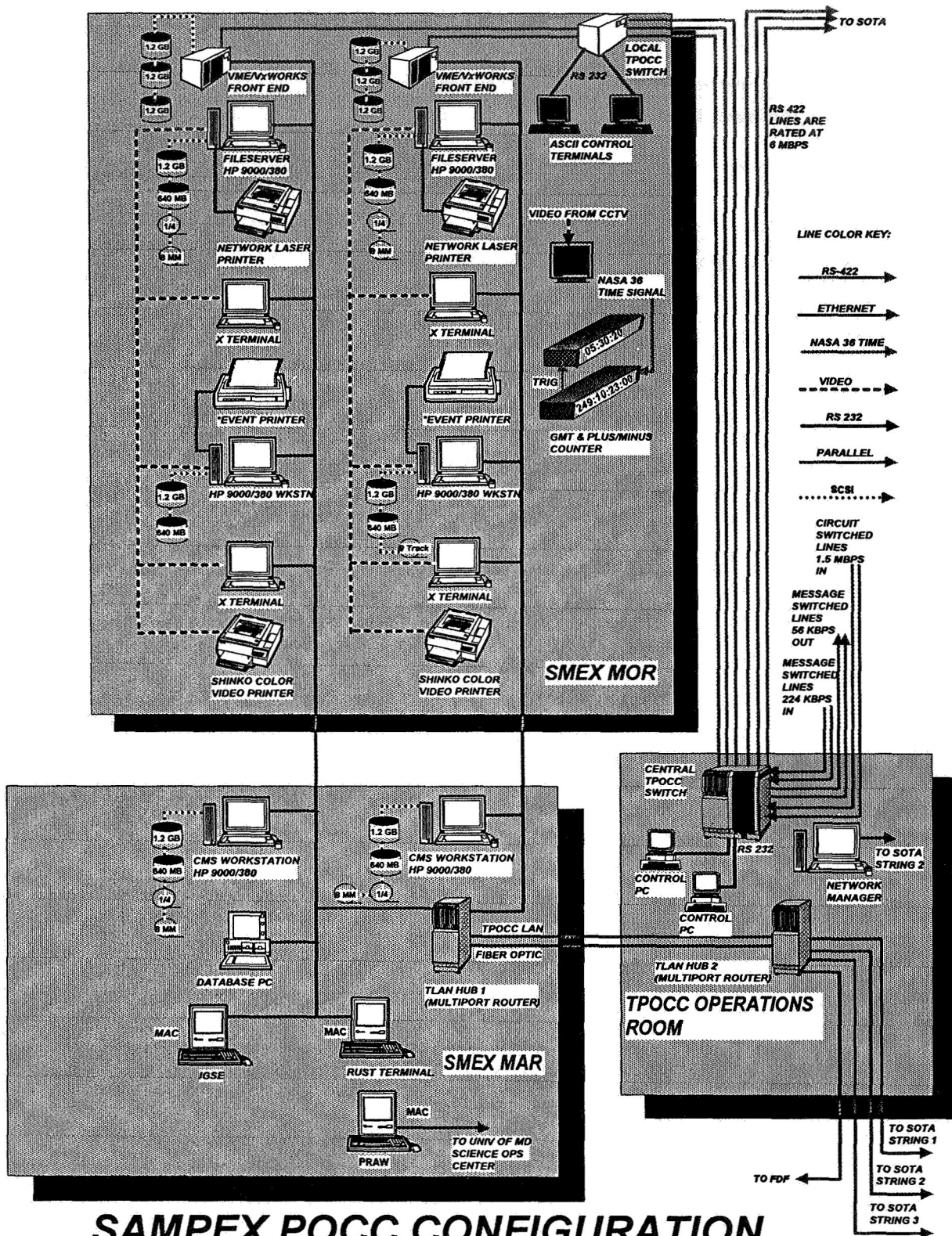
with each string having two operator positions. In addition, SAMPEX required three additional strings with 4 operator positions each which was provided in the launch and early orbit support facility called the Special Operations and Test Area (SOTA). During launch, each position was used by two people allowing 32 people to have display screens. The real-time functions reside in the front end with all of the user interface functions and some of the less time-critical functionality residing in workstations on the string.

Industry standards that were used in developing the POCC system were as follows:

- (1) Unix & C
- (2) X windows & MOTIF compliant software
- (3) VME based hardware for the real-time FEP
- (4) Ethernet, TCP/IP
- (5) Unify

The POCC functionality is partitioned between the FEP and the workstations. It should be noted that the FEPs run under VxWorks which is a real-time Unix-like operating system; whereas the workstations run under standard Unix. The following are the functions that run in the FEP:

- (1) State manager- manages and coordinates all of the real-time POCC functions and the different states they can attain
- (2) Events- logs and alerts operator of events that the operator needs to know such as the occurrence of errors in the data or the acquisition of data
- (3) GMT- inputs NASA 36 time for labeling of displays, reports and events as appropriate
- (4) Data services- distributes requested data to other processes on the network. Acts as data traffic cop. For example, a certain plot may request 4 parameters at 5 second intervals.
- (5) External Communications- Controls hardware which does NASCOM deblocking, frame synchronization and packet processing
- (6) Telemetry- decommutates the telemetry which resides in CCSDS packets and was received from the NASCOM deblocker and frame synchronizer and places the data in a memory current value table for access by the data server.
- (7) History services- records incoming NASCOM blocks and incoming transfer frames. Also allows for playback of this data.
- (8) Initialization & termination- Allows for start



SAMPEX POCC CONFIGURATION
Figure 1

up and termination of front end across network.

- (9) Command- Accepts and formats real-time commands and command loads for uplink

Each workstation on a string hosts the remainder of the functionality as follows:

- (1) Display- Provides displays of telemetry, telemetry status, command status, image dump status
- (2) TPOCC System Test and Operations Language (TSTOL)- Provides high level English commands for operator inputs. Also allows privilege classes and procedure generation.
- (3) File management- Manages files on workstations
- (4) Database- Accepts project database and transform into operational data base.
- (5) Reports- Generates required reports for users
- (6) Initialization and termination- Start and stop workstation

6. TESTS APPROACH:

The testing methodology of the SAMPEX POCC system was modified from the traditional approach due to time constraints. Typically, testing occurs in four phases: unit testing, system testing, acceptance testing and mission readiness testing. Typically, each phase might take 4-6 weeks. For the SAMPEX POCC, these phases were overlapped. For instance, the acceptance testers and the mission readiness testers received access to a system release about half way through system testing. This not only allowed early support of spacecraft testing but also served to provide feedback on problems in the system earlier in the development cycle.

The main test tool for the developers and the acceptance testers was the TPOCC Advanced Satellite Simulator (TASS). TASS is a TPOCC based simulator that outputs simulated telemetry data and accepts commands. TASS runs on the same type of FEP and workstation used by SAMPEX, so no additional equipment was required.

The FOT and the mission readiness testers used the Portable Satellite Simulator (PSS) and the actual spacecraft for testing. The PSS was a PC-based system that functioned very similar to TASS. The PSS was developed independently by a different group within the MOD.

Surprisingly, this compressed testing schedule resulted in software releases with less discrepancies. Most likely, this was due to the fact that the time criticality caused all users to look at the system in a more timely manner. In fact, launch was supported with two official releases and a few patches as compared to the four delivered for more traditional missions. We hope to use some of these tighter coupling concepts (minus the increased stress) between the developers and the testers in future SMEX missions to increase productivity.

7. FACILITIES:

The real-time operations area was housed in the SMEX Mission Operations Room (MOR). The off-line analysis area and the Command Management System (CMS) workstations were housed in the SMEX Mission Analysis Room (MAR). The SOTA served as a launch and early orbit support facility to accommodate all of the extra personnel who attended launch. The total space requirement for real-time operations (MOR) was less than 700 sq. ft. The MAR occupies another 680 sq. ft.

8. LESSONS LEARNED:

The main lessons learned in developing the SAMPEX POCC concerned what it takes to produce a POCC for a mission at lower cost and faster turn-around time. Although a large part of reducing cost thus far has been accomplished through the use of COTS software and hardware and the creation of reusable TPOCC software, the lowering of costs in the future will probably occur primarily through some restructuring of the management and development process.

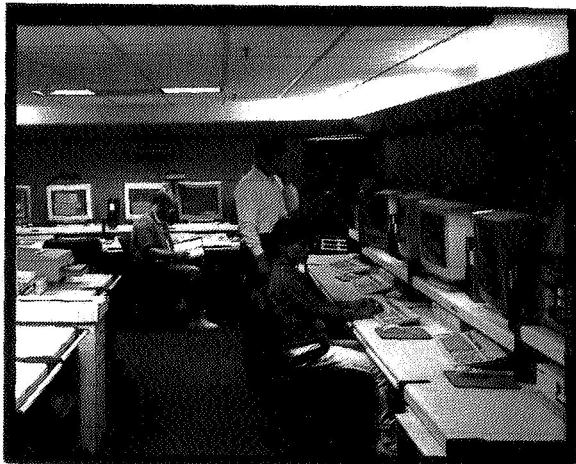
The goal of TPOCC has been to develop a core functionality which represents at least 80% of the functionality required by each new POCC. It is estimated that there will be approximately 85% reusability for the second SMEX mission which is Fast Auroral Snapshot Explorer (FAST). For the SAMPEX POCC, developers used about 60-70% of reusable TPOCC code. SAMPEX realized about a 25% cost reduction over traditional CCSB POCC development efforts. Continued cost reductions on future SMEX satellites will depend less on the amount of increased reusable software and more on the following:

- (1) Cooperation between users and developers in specifying requirements, in particular the

- streamlining of the requirements gathering process
- (2) Reduction of traditional presentations and paper to bare minimum and only to specify differences from previous missions in the series
 - (3) Additional streamlining of testing
 - (4) Continued emphasis on teamwork between the various contractor and government personnel

Some of the ideas to be incorporated for additional streamlining of testing are:

- (1) Early involvement of the testers in the development process so that the testers can help to catch problems earlier in development phase. The cost of resolving a problem is inversely proportional to how early in the development phase it is cost. An error caught during a design walkthrough has minimal impact, whereas an error caught on an operational release involves the effort of a few people to produce an audit trail, fix the actual problem and to retest the system.
- (2) Early involvement by the testers will also allow input to the developers as to the testability of the system.
- (3) Early involvement by the testers will allow their input into our primary test tool which is TPOCC Advanced Satellite Simulator (TASS).
- (4) Early involvement of the flight operations team and the mission readiness testers in the development process. Their detection of errors can be different since some of their discrepancies reported involve the actual operations of the satellite. As the actual users of the system, the FOT provided a different insight into using the system.



SAMPEX MISSION MOR

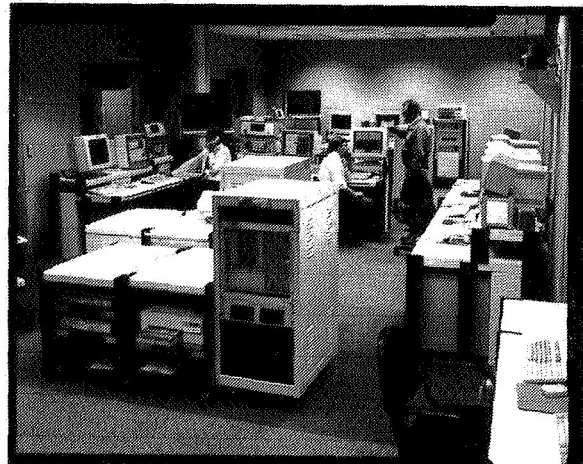
- (5) Test as much as possible in the actual operational environment. Seemingly small differences in environment became magnified during the stress of prelaunch system checkout.

9. FUTURE ENHANCEMENTS:

Future enhancements to the SMEX MOR will allow for multiple simultaneous satellite supports. The SAMPEX MOR contains two POCC strings, one designated as the primary string and one as backup. A third string will be added for FAST, the second SMEX satellite, giving each mission a dedicated primary string and a shared backup. When SWAS, the third SMEX satellite, is launched, the three POCC strings will need to be shared among the three SMEX satellites. One scenario for supporting all three satellites is to dedicate each string to one satellite, but also to allow any string to provide backup support for any satellite. This scenario will be accomplished by scheduling of the equipment and carefully planning operations

Also, with most of the core POCC functionality complete, the TPOCC group will focus on adding enhanced user interface tools and off-line processing capabilities. Some of the tools being developed both internal and external to the CCSB are:

- (1) GenSAA (previously mentioned)
- (2) Space Views, a three dimensional visualization tool for flight operators and analysts
- (3) Orbital Signature Analyzer, pattern recognition tool to monitor the health and safety of a spacecraft by analyzing the shape of a plot
- (4) Trend analysis tool box- Off-line tool to perform statistics calculations and trend



SOTA USED FOR L&EO

A USER VIEW OF THE EURECA GROUND SEGMENT

Paolo Ferri and Steve J. Kellock

N94-23844

European Space Agency / European Space Operations Centre
Darmstadt, Germany

ABSTRACT

EURECA, the European REtrievable CARrier was launched on July 31st 1992. It is the largest ESA spacecraft ever launched, and the first one to be launched and retrieved by the NASA Space Shuttle.

The many new aspects of this mission affected the operations concept and the ground segment design in all important areas: an off-line concept for mission control has been applied, based on automatic commanding and post-contact telemetry analysis; mission planning is the centre of all routine operations activities using dedicated tools and operational techniques; high precision orbit determination and daily update of related telecommands for spacecraft control are needed to cope with the requirements coming from the low altitude orbit and the spacecraft attitude control design.

The paper describes the lessons learned during the first months of utilisation of the EURECA ground segment from the point of view of the flight control team.

Key Words: Mission control, operations, ground segment

1. INTRODUCTION

EURECA, the first European spacecraft designed to be retrieved and re-used for subsequent flights, was launched on July 31st 1992 on the NASA Space Shuttle Atlantis. After deployment it was manoeuvred to an operational circular orbit at 508 Km altitude, where it will carry out scientific operations for its fifteen payload instruments, mainly in the field of microgravity research and space science. After a planned mission of nine months the spacecraft will be transferred to a retrieval orbit, where it will rendez-vous with a new Space Shuttle, which will retrieve it and bring it back to Earth.

The EURECA ground segment was designed around the main mission characteristics of reduced visibility time and high level of spacecraft autonomy, large number of different possible payload operational configurations and packet telemetry and telecommand concepts. It consists of two ground stations in Maspalomas and Kourou, which provide about eight contact periods of 5 to 10 minutes each every day, spaced by one orbit which lasts about 90 minutes (a long non-coverage period of about 9 hours occurs

daily between two sequences of station passes); and a control centre located in Darmstadt, which can also make use of a third station in Perth as a back-up.

For the periods when EURECA was attached or in proximity of the Space Shuttle contact with the spacecraft was established via the NASCOM network, the NASA TDRS system and the Orbiter itself used as a data relay station. In those periods, which include the first two days and the last few days of the mission, the contact with the spacecraft is practically continuous, and a completely separate telemetry and telecommand interface, as well as a different way of operation had to be defined.

This paper does not describe the EURECA ground segment and control system, but rather the direct operational experience accumulated with the different parts of it and makes suggestions for possible future improvements.

2. MISSION CONTROL SOFTWARE

2.1 Database Editors

The EURECA operational database was built manually using the manufacturer's spacecraft database developed for the system level testing activities, complemented by information collected in a large number of design documents. More than 8000 telemetry parameters, 2500 telecommands and 4800 report messages had to be defined, an enormous task in terms of time and manpower both for creation and later maintenance of the files.

The editors used for this task within the EURECA Dedicated Control System (EDCS) were also constraining this work, not providing the necessary level of flexibility, in particular when large changes to the source database had to be introduced. The solution of automatically importing into the operational database the industry developed spacecraft database would have helped in the traditional areas like housekeeping telemetry definition; the flexibility given by the EURECA packetised telemetry and telecommand concept would have been however significantly constrained if the entire database definition had been left to the industry.

A mixed solution of general database information imported directly and later

processed using editors which are more change-oriented rather than input-oriented would probably satisfy all the needs of such a complex database generation task.

2.2 User Interfaces

The user interface for most of the tasks provided by EDCS to the EURECA flight control team is provided on workstations with very limited graphic capabilities, low speed of interaction with the central computer, and in general reduced flexibility in the use of the three available displays.

For a complex mission like EURECA the standard spacecraft monitoring and commanding tasks controlled via the workstations require more display space for command building and displaying of the different types of telemetry; in addition several tasks related to ground data management and interface with the ground stations have also to be carried out by the spacecraft controller using the workstation. This resulted in an increased need of display availability, which can hardly be satisfied by the arrangement provided by the workstations. The limited graphic capabilities impact in particular the off-line data analysis carried out daily by the flight control engineers.

The combination of workstation and user interface limitations also practically prevented the creation of a database of mimic displays, normally very effective in summarising information and thus allowing savings in space and time.

Interactive generation of telemetry displays is provided for graphic and alphanumeric scrolling displays. This was extensively used during system and spacecraft tests, and it is still found very useful for quick-look analysis of unexpected spacecraft behaviour during flight. This type of features should be extended to all types of telemetry displays, from alphanumeric to mimic; the system should also allow a direct consolidation of the changes performed interactively at the workstation into the operational database without going through the editors.

A transition to modern, window-oriented workstations is taking place at ESOC. The use of the currently existing application software via the new workstations has already proven to sensibly improve the system effectiveness. A combination of task-dedicated displays and windowed displays is considered to be the optimum solution.

2.3 Telemetry Processing

The telemetry processing system makes full use of a new system software designed to handle packetised telemetry. Different tasks had to be designed to cope with the

NASCOM interface for the mission phases when telemetry is routed via the Space Shuttle through the NASA communications network, and the ground station interface for the other phases of the mission.

Standard telemetry processing features are provided, like limit checking, validity and derived parameters calculation for the housekeeping telemetry packets. Other types of packets are handled according to the type in different ways.

The problems experienced with the telemetry processing system in this first part of the mission are all related to the way the system reacts to corrupted data received from the spacecraft. The large number of independent processors on-board EURECA increases the likelihood of unexpected behaviours which result in corruption of the format or contents of the TM packets produced. Very strange anomalous behaviours have been observed in some of the payload processors, like position shifts of packet time field or overflow in the packet counter, which caused serious problems to the ground software, ranging from continuous alarm generation to crashes of the telemetry processing tasks. In most of the cases ad-hoc software patches have become necessary on-ground to cope with the new or sporadic anomalous situation.

The design of a telemetry processing system has to be based on some assumptions on the structure of the data to be processed, and is therefore particularly vulnerable when the perverse behaviour of an on-board processor corrupts the data in a specific and unexpected way. For the same reason, however, the system should be flexible enough to allow a rapid configuration of the telemetry processing software in order to adapt it to the new situation in case of an on-board failure. The system should for example allow the user to disable specific checks on selected packets, to modify the time calibration and filing rules for specific packets, without the need of software modifications.

2.4 Telecommands Handling

Three parallel command queues are provided for EURECA commanding during a ground station pass: the manual commanding queue, which allows real-time commanding with direct control on single telecommands or timed sequences; the pass schedule queue, which can be started in the background and executes a series of pre-configured commands at specified times relative to the start of the queue; the maintenance queue, which allows uplink of all the previously prepared time-tagged commands to be stored on-board for later execution. This arrangement allows the spacecraft controller to concentrate on the manual commands, leaving the control of the background automatic queues to the system; he is in control of the start and stop of the

command activity via the background queues.

Very useful has proven to be the capability given to the user to create and store manual stacks of commands, which, together with the command files editors for pass schedules and master schedules provide the required flexibility to operate EURECA in the routine off-line way, but also to efficiently react in a short time to unexpected real-time commanding needs.

When problems in the command link are experienced, however, the normally very smooth commanding becomes extremely difficult to handle. In particular a better visibility of the commanding status for the different queues would be needed, including a direct presentation of spacecraft and ground station messages to establish the command status. Handling of uplink failures could also be improved by automating analysis tasks which are currently carried out manually whenever failures or interruptions occur during the uplink of automatic queues.

2.5 Automatic Command Verifier

One of the most useful and widely used software tasks in the EURECA control system is the automatic command verifier. The complexity of telecommand routing on-board EURECA and the variety of responses the spacecraft can generate which can be used to derive the success in telecommand reception and execution is handled by a single task. Based on user-defined rules contained in the definition of each telecommand in the operational database the command verifier task accesses all telemetry streams and summarises the telecommand verification result as one of 23 different states, ranging from complete success to complete failure; the resulting status is recorded in a telecommand history file.

The telecommand history of the previous 24 hours is analysed daily by an engineer, who concentrates his investigation only on those few commands which were not completely successful. Manually scanning through one day of telecommands - typically for EURECA routine operations this means about 800 telecommands - takes only a few minutes if all telecommands are successful. On the contrary, the investigation of the reasons for the failure of only a few telecommands can take a significant amount of time.

Possible improvements to help speeding up the trouble-shooting activities related to a failed telecommand could be in the area of automatic selection of telemetry information which is relevant to a selected telecommand; also an extension of the verification task to allow it to follow in telemetry the entire process initiated by a telecommand, and not only the successful start of the process, is

under investigation. This is particularly feasible on a spacecraft like EURECA, where the high level of on-board automation results in long duration automatic processes, lasting even several days without intervention from ground; the process controllers produce periodic or event-driven messages in telemetry which give an indication of the progress of the activity.

2.6 Event Messages Handling

All payload instruments and most of the spacecraft subsystems on EURECA generate special telemetry packets, called event packets, which report asynchronously the success of a telecommand, the start, end or progress of an automatic process, hardware failures or any other unexpected event detected on-board. These packets are used in the telecommand verification process, but can also be displayed on a scrolling display which shows for each packet a fixed, user-defined text message for each packet. Many of these packets also contain housekeeping-like parameters: a typical example is a snapshot of the entire housekeeping telemetry of a payload instrument, generated by the instrument processor only at the time of a relevant event. This approach, adopted by several EURECA instruments, makes the best use of the packet telemetry concept, generating telemetry information only when it is necessary, thus avoiding high frequency housekeeping telemetry sampling and saving space in the downlink. Parameters contained in the event messages can be displayed as part of the text message attached to the packet and calibration or text interpretation can be applied to them in the same way as for normal housekeeping parameters.

However event packets parameters are still treated separately from the standard housekeeping parameters, and therefore not integrated in the rest of the telemetry processing, like limit checking, validity and derived parameters calculation. As there is in principle no difference from the on-board processor's point of view between housekeeping parameters transmitted in an event telemetry packet or in a standard housekeeping packet, this limitation of the EURECA telemetry processing system is arbitrary and causes some difficulties, in particular for those payload instruments which base their telemetry generation on the above described event-driven concept.

For future autonomous spacecraft event-driven reporting will most likely become more and more common; a full integration of event packet parameters in the telemetry processing chain will therefore be mandatory.

With its limitations the event messages display task remains nevertheless one of the

most extensively used tools for monitoring of the on-going EURECA activities.

2.7 Flight Dynamics Interface

The role of flight dynamics software in routine mission control of EURECA is very important: the low altitude orbit requires daily updating of orbit determination and predictions; on-board attitude control and ground mission planning software also require frequent updates of orbit information, to keep attitude and planning accuracy within the specified requirements. In the first weeks of the mission a dedicated flight dynamics team was in charge of generating the necessary orbit and attitude related products using software tools running on a dedicated computer. With the progress of the mission and the start of the routine operations phase the flight dynamics team, which includes an independent quality control group in charge of verification of the generated products, has been gradually reduced with the target of leaving the routine flight dynamics tasks to a set of automatic routines which are started daily by a scheduler task. These routines perform all required tasks from collection of tracking data received from the ground stations and orbit determination to orbit prediction and related products generation, including orbit model telecommands to be transferred to the front-end computer for uplink to the spacecraft.

A weak point in this scheme is that telecommands are automatically generated which cannot be easily checked by the flight controllers in charge of uplinking them to the spacecraft. Any problem in the automatic generation software, which can also be caused by corrupted input data, is not detected any more by the quality control check carried out in the first part of the mission. The only protection the current system provides is limit-checking on telecommand parameters. This is however only effective on a limited number of parameters and by no means represents a complete check.

Some simple independent software checking is being implemented to trap any major problem with the automatically generated telecommands. A more consistent software, possibly based on the same routines which were developed and used by the quality control team during the early phases of the mission, should be implemented for future missions and integrated in a more comprehensive telecommand generation software.

3. SCIENCE OPERATIONS

3.1 Mission Planning

One area where the existing workstation interface had to be abandoned years ago is the mission planning task. This tool runs on

an independent VAX workstation where the graphical and windowing capabilities are used in producing an effective user-interface.

On the other hand the mission planning is probably the area of the mission control software which is experiencing the most significant problems. The tool accepts scheduling inputs based on orbital constraints and, using a pre-defined database, it produces a resource consumption profile for all spacecraft resources available to the payload (eg. power, data storage, cooling capacity) by adding up the contributions from the single scheduled operations. If the total resource consumption goes above a pre-defined threshold the mission planning tool warns the user that a 'clash' is detected, allowing him to modify the payload operations schedule to solve the problem.

Experience has shown, however, that the type of operational constraints to be applied to payload operations scheduling are more complicated than simple relations to orbital events. Payload operations are often constrained by relations between activities to be executed on the same or on different payload instruments, ground activities, specific requirements by the investigators. This type of constraints is not handled by the mission planning software, and a combination of manual scheduling and user-developed software had to be adopted to simplify the actual planning tasks. An additional tool would be required to allow the planner to specify and modify rules to be used by the mission planning task to schedule the required payload operations. This tool should be flexible enough to allow specification of rules which are normally thought of or required by the investigators or by new developments in the spacecraft situation only later in the mission plan preparation and often even during the actual mission.

Resource consumption checking and clash detection handling turned out to be less critical than expected in the actual planning exercise: planners very soon acquire enough experience on possible payload configurations which allows them to manually produce feasible payload operations timelines. A lot of development effort was therefore put into a less important part of the software task.

Another problem experienced with the available tool is the little visibility the planner has on the already scheduled operations. This is in particular important when changes have to be implemented in already scheduled operations due to resource availability problems, new failures or new requests from the investigators. An analysis tool which allows to identify at any point of the scheduled timeline what payload operation is contributing to the overall

payload configuration would be a significant improvement required for this task.

3.2 Data Disposition System

The science data generated by the payload are distributed to the users via electronic means: all consolidated spacecraft data are accessible from the users via a separate computer, which is linked to the operational machine and to the external world via different communications protocols for security reasons. A catalogue of available data can be requested via electronic mail by the user, who can then specify the appropriate request for data from his own instrument. Available to the user are also all spacecraft housekeeping data and other ancillary information files like future orbit events, attitude history or telecommand history.

The user can also specify, via the same electronic interface, requests for special operations to be conducted on his instrument, or changes to the pre-mission defined operations plan. This type of request, called TC Request, forms the input, together with the baseline payload operations plan, to the daily mission planning exercise.

Unfortunately the loop with the user is not closed electronically: TC requests cannot be input directly in the telecommand scheduling process but the related telecommands have to be manually generated by a mission planning engineer. A more automatic handling of TC requests was not possible due to the large number of payload instruments and different types of operation requests, together with the limited time and budget available for the development of this system.

A first necessary improvement of the interface with external users would be to allow the mission planner to include electronically the approved TC requests in the telecommand generation process. Automatic checking and approval of TC requests, allowing each user to remotely control his own instrument independently of the other spacecraft operations is the target, still far away from the EURECA approach, but at least visible in the distance.

4. GROUND STATIONS - CONTROL CENTRE INTERFACE

4.1 Telemetry Interface

Real time and on-board recorded telemetry is received and stored at the ground station by a telemetry frame pre-processor; During a telemetry dump the data rate reaches 256 Kbps and due to the lower capacity of the station to control centre link only a subset of the received data can be transferred in real-time. The control centre can pre-program the station unit to transmit any selection of

telemetry packets or all real-time or recorded telemetry frames as received from the spacecraft.

During the pass only real-time telemetry is normally transferred directly to the EDCS, whilst the telemetry which is dumped from the on-board memory during the pass is transferred in the non-coverage period between two passes.

Whilst the nominal data transfer is adequately handled by EDCS, the problems start in case of interruptions during the data dump from the spacecraft or during the data transfer from the station.

If the problem was on the ground link the system allows the user to initiate a new transfer; unfortunately in this case a re-transfer of the entire selected data set is performed, and not simply of the data lost on the link. Due to the long duration of the transfer of all data dumped during the pass this becomes a very inefficient and time-consuming operation.

If the spacecraft-station link is interrupted during a telemetry dump the problem is more serious and difficult to detect and to recover. The system allows in fact detection and later filling of gaps created in the telemetry history files by any link interruption; however the tools available for this task are very complicated to use and require long manual investigations and calculations to determine where and when the problem occurred and what data have to be re-transferred. In many cases this time is of the same order of the wraparound time of the on-board memory, making the data recovery physically impossible.

A more efficient and user-friendly tool would be required to allow immediate detection of the gap, identification of whether the data were lost on the space-to-ground link or between the station and the control centre, and indicate what the recovery action should be. The necessary information is available on ground to completely automate the task, leaving to the operator only the decision whether to initiate the recovery process or not.

4.2 Telecommand Interface

Experience during the mission with the telecommand interface between EDCS and the ground station has been very positive, with hardly any problem occurred in more than three months and more than 80000 telecommands uplinked to the spacecraft.

Problems occurred in the development phase, due to the late decision to close the telecommand block protocol loop with the spacecraft at the control centre, and not, as initially foreseen, at the ground station. This

increased dramatically the complication of the command uplinker software, which has to cope for every telecommand with a number of messages coming from different units at the station in addition to the telemetry messages from the spacecraft.

Testing this software in a realistic environment became absolutely necessary due to the importance of the timing aspects of the problem and this forced extension of precious testing time with the spacecraft flight model connected to a ground station interface.

For the NASA interface a complete realistic test was not possible, leaving the fine tuning of several configuration parameters to the actual flight experience.

Fortunately no correction to the uplinker software became necessary during the mission. It is however advisable, in order to simplify significantly the command interface software at the control centre, to close as much as possible of the space to ground loop at the ground station.

4.3 Tracking Interface

The interface with the ground station which deals with tracking data collection from and antenna pointing information transmission to the station was given a low priority in the software development phase, resulting in a relatively simple implementation.

EURECA tracking data are collected at the station by a unit called MPTS, which is remotely programmed for operation and data transfer from the control centre. The only problems experienced are also in this case in the area of the user interface, which gives little visibility to the spacecraft controller of the status of the unit and in particular of the contents and status of the programmed queues.

Antenna pointing information is generated by the flight dynamics automatic software, and transferred to the ground station by the network operators on a daily basis. In this case too, the operator in charge of the transfer has no visibility of the contents of the files he is transferring, and only when the data reach the station any problem that may have occurred becomes available. A recurrent problem in the first part of the mission was in fact that antenna pointing data were not reaching the station in time, or the station would request new data which were not yet available.

This type of visibility problems are usually worked around by experience and procedures improvement in the first months of the mission. However, a more elegant interface which links directly the flight dynamics generation software to the ground station computer and performs the data

generation or transfer automatically or on request would be a significant improvement in this area.

5. CONCLUSIONS

The EURECA ground segment has to date successfully supported this challenging mission for almost half of its nominal lifetime.

Spacecraft routine operations still keep a team of eight spacecraft controllers and eight engineers extremely busy; the ground segment is however helping to carry out the daily tasks in time, and this is also shown by the fact that we could afford the time to write and present this paper.

The experience gained with the EURECA mission control should be used to improve for future missions the ground segment reliability and to reduce the involvement of man in all those tasks which can in principle be automated.

DEVELOPMENT OF THE CASSINI GROUND DATA SYSTEM IN A MULTIMISSION ENVIRONMENT

N94-23845

G. Madrid and G. Wanczuk

Jet Propulsion Laboratory, California Institute of Technology

ABSTRACT

As baselined, the Cassini Ground Data System (GDS) will be composed of Project-specific and multimission elements. The former will be developed by the Cassini Project and the latter by two JPL institutional organizations, the Telecommunications and Data Acquisition Office (TDA) and the Multimission Operations Systems Office (MOSO).

The GDS will be developed in three principal phases : Spacecraft Test, Launch-cruise, and Science Tour, with a significant part of the development deferred until the post-launch period. New capabilities are being introduced that are key to the achievement of more cost effective operations .

Successful development of the system will require careful planning and will involve participation of diverse disciplines. This paper introduces the Cassini Project from the Ground Data System perspective and discusses development approaches expected to produce systems which meet functional and performance requirements and which will be delivered on schedule and within budget.

Key Words : system development, system adaptations, ground data system, system integration, multimission systems, testing of multimission systems.

1. INTRODUCTION

Mission Description

The Cassini Project will launch a spacecraft on a trajectory to Saturn in 1997 for the purpose of conducting scientific investigations of the planet, its composition and magnetosphere, its satellites, and its ring system. The spacecraft will carry a payload of 12 instruments and a European Space Agency (ESA) provided Titan probe which will carry 6 instruments.

Scope of GDS

The Cassini Ground System (GS) consists of the teams, procedures, hardware, software, and facilities required to operate the Cassini Mission. The Cassini Ground Data System (GDS), which is the topic of this paper, includes the hardware and software parts of the Ground System (see fig. 1). Requirements definition for the GS is now near completion and the design phase will soon be underway. The overall GS architecture will be selected early in 1993; the GDS described in this paper represents a subset of the baseline Ground System (GS) architecture.

Development Objectives

The overall development objective is to produce a ground data system which satisfies the negotiated requirements on schedule and within budget. The development is constrained by the requirement to phase as described in section 1.4 below and must include new capabilities to facilitate reduced operations costs.

Inheritance of mature software used by existing projects together with experience gained in developing multimission software and adaptations for those projects will substantially contribute to the achievement of the development objectives. Innovative variations introduced into the Integration and Test (I&T) process are intended to result in well-tested, stable deliveries to operations.

Phased Development

Budgetary constraints require that the GDS be developed in three phases : Spacecraft Test, Launch-Cruise, and Science Tour (see figs. 1 & 2). The Spacecraft Test phase will include capabilities to process telemetry data and to generate simple sequences and commands while the spacecraft is at JPL or remotely located for launch preparation . The Launch-Cruise phase will include additional capabilities to track the spacecraft, acquire data, radiate commands, and monitor the spacecraft's health. The Science Tour phase

will include additional capabilities to process and archive science data, to generate and validate complex sequences, and to perform spacecraft analysis functions with the aid of more automated software tools. All phases will include remotely-located Science Operations Planning Computers (SOPC's) which will provide processing and central data base access capabilities to the investigators.

2. GDS IMPLEMENTATION

The GDS will be composed of multimission components provided by the Multimission Operations Systems Office (MOSO), project-specific components provided by the Cassini Project Mission Operations System (MOS), and other multimission components provided by the Office of Telecommunications and Data Acquisition (TDA). This paper will focus on the MOSO and Project-provided capabilities because the main characteristics of the GDS architecture are associated with these organizations; the TDA capabilities are contained within the TDA's separate, stable, standardized architecture which has proven to be relatively independent of project variations.

TDA

The TDA-provided capabilities are delineated in Figure 1 and include the following :

Telemetry

- acquire telemetry signal
- extract data/bit synchronize

Command

- receive command information
- radiate command signal

Radio Metric

- acquire doppler data signal
- digitize signal

MOSO

The MOSO-provided capabilities are also delineated in Figure 1 and include the following :

Telemetry

- frame synchronize
- decommutate, channelize

Command

- translate to binary

Simulation

- spacecraft telemetry
- command interface

Sequence

- generate, validate, translate

Image Processing

- real time process
- systematic process analysis

Multimission Spacecraft Analysis

- monitor health and safety
- analyze trends
- assess performance
- produce predicts
- analyze anomalies

Mission Analysis and Design Tools

Science Planning Tools

MOSO development of multimission "core" capabilities is planned and carried out generally on annual cycles. Specific delivery dates are negotiated with affected projects. Capabilities proposed for inclusion in a software version are reviewed with MOSO entities and affected projects; final decision is made by MOSO.

Actual development is performed to MOSO requirements by personnel in JPL technical divisions. Reviews will be conducted consistent with MOSO standards.

Documentation of core developments is the responsibility of MOSO and is performed to MOSO standards.

Adaptation of MOSO core software is usually the responsibility of the affected project. One exception is the first time development of a MOSO core capability, in which case MOSO may also develop the adaptation.

Project-Specific

Project-specific capabilities fall into two categories: adaptation of MOSO core software and project-developed software which is independent of the MOSO-provided software. This latter category is sometimes termed "stand-alone" software.

Adaptation of MOSO core capabilities is required due to lack of standardization of spacecraft information systems design and the requirement to tailor operational interfaces to accommodate project needs. Adaptation will be applied to MOSO telemetry, command, sequence, image processing, spacecraft analysis, and navigation subsystems, as well as to science

planning tools.

The required adaptation is performed by personnel in JPL technical divisions, often by the same personnel who developed the core capabilities. Reviews will be conducted consistent with project standards.

Documentation of adaptation capabilities is the responsibility of the project and is performed to project standards.

Project-specific stand-alone software will be diverse and will include models and analysis software. This software will be developed to project requirements by project personnel or by personnel in JPL technical divisions. Reviews will be conducted consistent with the project software management plan.

Documentation will be produced consistent with the project software management plan.

3. NEW FEATURES

The Ground Data System will include several required features that are needed by the Cassini project and MOSO partly to facilitate reduced operations costs. Among the most important of these features are: the Test Telemetry and Command Subsystem (TTACS), the Multimission Spacecraft Analysis Subsystem (MSAS), and the High-Speed Spacecraft Simulator (HS-SIM).

TTACS

The Test Telemetry and Command Subsystem (TTACS) feature is a cost-saving, risk-reducing item which will support the spacecraft Assembly, Test, and Launch Operations (ATLO) phase. It is not new technology but a repackaging of the basic downlink and uplink capabilities of the multimission system, adapted for Cassini, into a portable set of two or three workstations. The TTACS can be used to support spacecraft assembly and test operations at remote JPL locations, and can later be moved to Kennedy Space Center to support the test and launch operations at that site. This capability will greatly improve the effectiveness of the Spacecraft Integration and Test Team by permitting them to control their own test string while maintaining datalinks to the main flight string at JPL. In the past, the team had either to do their testing on support equipment, completely separated from the flight control

environment, or to contend with developers and other testers for time on the flight test string. Development will also benefit because the TTACS provides early experience with the Cassini-adapted multimission system and accordingly permits an early shakedown of capabilities. Operations will benefit because the user interface will not substantially change from phase to phase, thus reducing training time and operational errors.

MSAS

The Multimission Spacecraft Analysis Subsystem (MSAS) is an entirely new development which integrates all of the spacecraft subsystem engineering analysis (including spacecraft health and safety) functions into one system with a shared data base. It provides for sharing of data and correlation of events between spacecraft subsystem analysts. Besides providing an integrated, distributed, computing environment for these functions, MSAS will also provide a set of tools which will automate many analysis tasks, thus diminishing the time and effort required to analyze and react to problems on-board the spacecraft.

MSAS will be developed using the Rapid Development Methodology which will consist of multiple deliveries each building on earlier deliveries.

MSAS is intended to reduce the size of the spacecraft teams during operations and to reduce development costs for subsequent projects. Because it replaces necessary flight control capabilities previously developed by projects, it is critical to the Cassini Mission.

HS-SIM

The software-based High Speed Spacecraft Simulator (HS-SIM) is required to satisfy the Sequence and Spacecraft Teams' requirements for a fast turn-around simulator to validate command sequences before they are radiated to the spacecraft and to support spacecraft analysis activities. Through the utilization of relatively inexpensive microchip technology, the multimission High-Speed Simulator will provide performance rates considerably in excess of real-time.

The HS-SIM will reduce operations costs and reduce risk in both the uplink and the analysis areas. In the past, simulators were

tailor-made for each mission, as required. The development of a multimission simulator will be advantageous not only to Cassini but to future missions as well, and will thus reduce overall development costs to NASA.

4. INHERITANCE AND EXPERIENCE

The initial operational version of the MOSO multimission system supported the Magellan Mission and a subsequent version is now supporting the Mars Observer Mission. Additionally, the Voyager, Ulysses, and Galileo projects are in the process of adapting the multimission system to support their respective missions, enabling the decommissioning of obsolescent data systems with substantial operations and maintenance cost reductions.

Because of this previous multimission GDS development experience, the Cassini project is obtaining a multimission system with substantial inheritance. Characteristic of all development efforts, there have been fewer problems with each succeeding version; the resolution of problems in the earlier versions adds to the stability of the succeeding versions. The version on which the Cassini GDS will be built will be derived directly from the version now in use by Mars Observer.

MOSO's multimission GDS development experience shows that the following major factors are detrimental to multimission product quality: (1) late requirements, (2) late definition of data interfaces, (3) late dependencies, (4) underestimation of required development and test resources, and (5) insufficient time for integration and test due to schedule compression resulting from the previous four problems.

In order to minimize problems in the Cassini GDS development, the following steps are being taken by the Cassini Project and MOSO: (1) timely specification of requirements and negotiation of commitments, (2) early definition of interfaces and establishment of system engineering working groups at all levels, (3) identification, tracking and follow-through on all commitments and dependencies through a receivable/deliverable reconciliation process, (4) frequent and thorough planning and review of all ground system schedules and resource plans, (5)

development of contingency plans for major delay scenarios, (6) establishment of a Project-MOSO management coordinating committee.

5. INTEGRATION AND TEST

One of the key processes involved in creating any successful ground data system is Integration and Test (I&T). When a multimission system is the subject of the I&T process an additional challenge is introduced because the multimission system elements must be adapted to meet project-specific requirements. An appropriate strategy must be selected to ensure that the combined adaptations and multimission system elements are properly integrated and tested. The following describes how this challenge is being dealt with in the development of the Cassini GDS.

Currently, the responsibility for the I&T process is divided between MOSO and the project test teams. MOSO is re-evaluating this current practice in light of the experience gained in integrating adaptations with multimission system elements. One proposal will be the formation of an integrated MOSO-Project test team to perform all of the System - level integration and test functions. The proposed test team would perform the relevant parts of the project activities known as User Acceptance Test and GDS Test. In this way the resources for testing will best be optimized to reduce overall cost.

The integration and test process for adapted multimission software is evolving as the multimission concept evolves. There are corresponding variations in the subsystem phase of adapting and integrating multimission system elements for the Cassini GDS; these occur because of the variations in the phasing of the adaptation with the core implementation. Figure 3 illustrates the three situations that occur most frequently in the I&T process. Upon analysis, we find that the three situations depicted in figure 3 are variants that occur due to the interleaving of the multimission and the project integration and test life-cycles.

The Cassini GDS will be implemented through a series of phased deliveries. The MOSO deliveries will be phased to negotiated need dates for specific capabilities. These phased deliveries must be integrated with

commitments to other projects that are also users of the multimission system elements. Scheduling of the Integration and Test of the GDS capabilities will be accomplished through a Project/MOSO coordination process.

6. CONCLUSIONS

The planned approach to development of the Cassini GDS is expected to enhance the probability of successful delivery of all required capabilities on schedule and within budget.

Requirements and interfaces are being defined early in the development lifecycle, consistent with development standards.

The multimission and project adaptation development expertise and the multimission software maturity achieved in the Magellan and Mars Observer developments will be of significant benefit to the Cassini development effort.

The Telemetry, Command, Sequence, Navigation, and Simulation subsystems will include significant inheritance from previous projects.

The Telemetry, Command, Sequence, and MSAS Subsystems will be developed such that there will be a series of deliveries with increasing capabilities; each delivery will build on the maturity of the earlier delivery.

Improvements in the test and integration processes, together with timely deliveries from development organizations, are expected to result in timely deliveries of well-tested capabilities to operations.

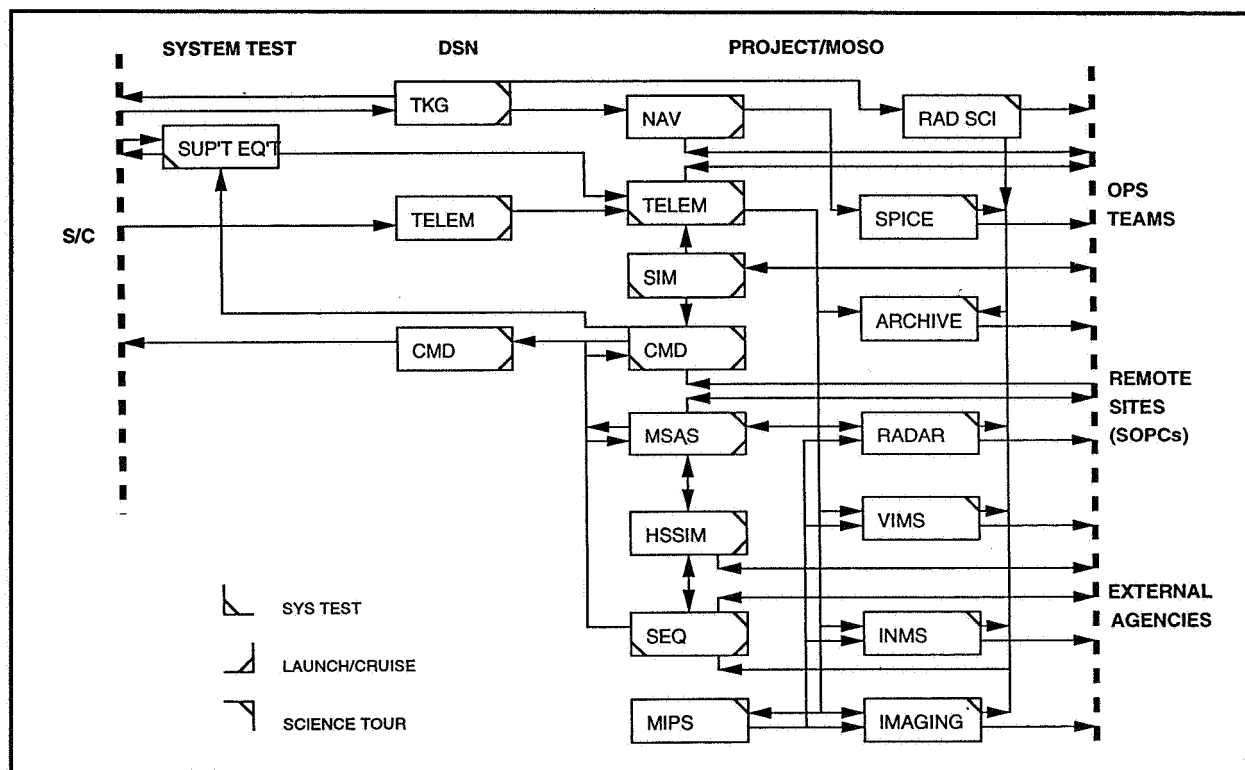


Figure 1. Diagram of the Cassini /TDA /MOSO Ground Data System

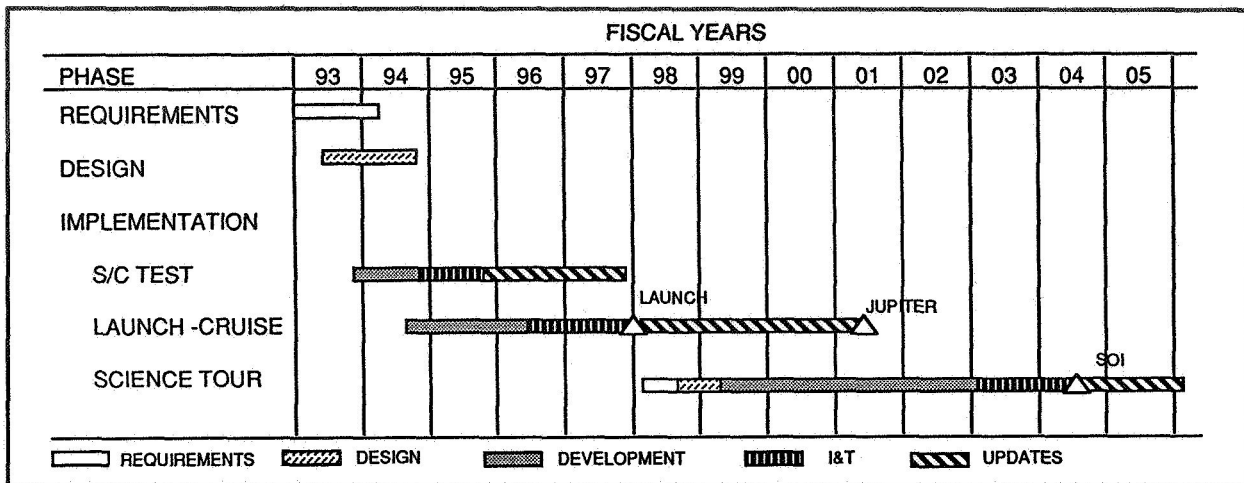


Figure 2. Phases and Principal Deliveries of the Cassini / MOSO Ground Data System

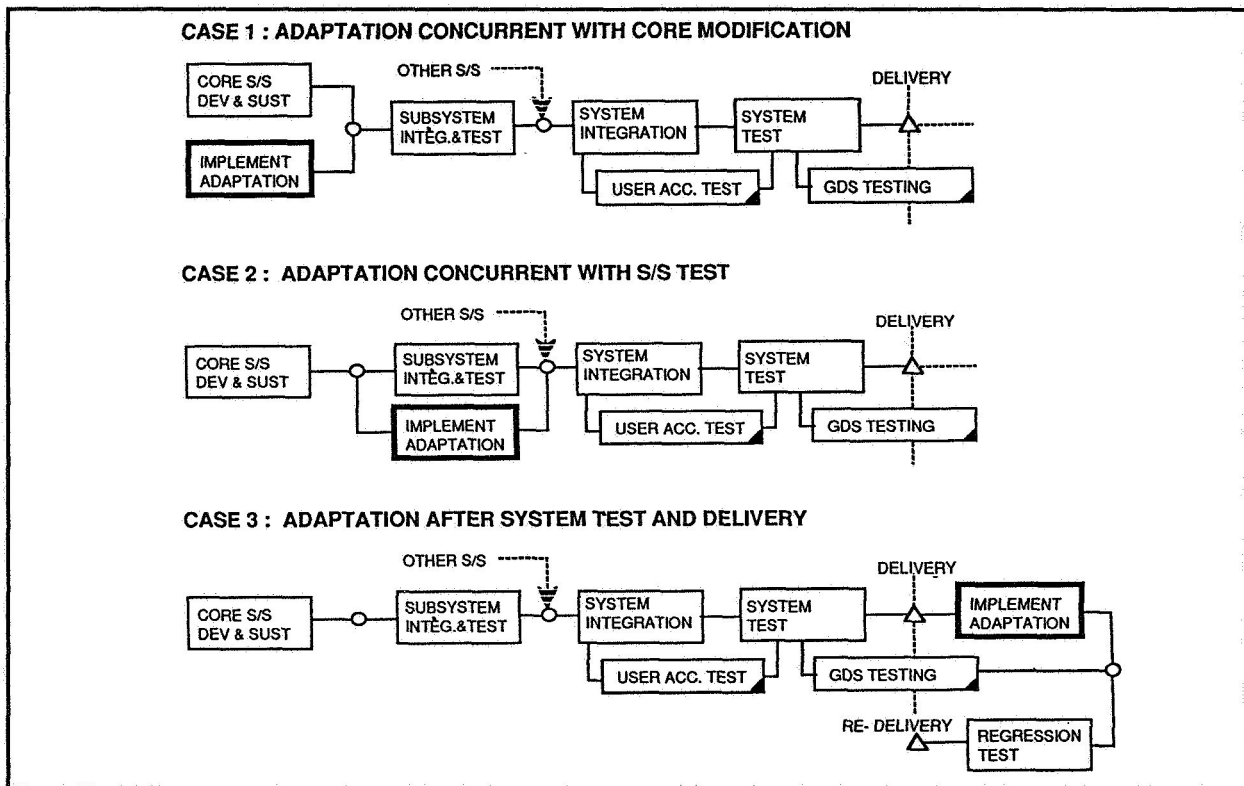


Figure 3. Three Basic Situations of Integration & Test of Multimission System Elements with Project Adaptations

SPOT SATELLITE FAMILY: PAST, PRESENT AND FUTURE OF THE OPERATIONS IN THE MISSION AND CONTROL CENTER.

Mr PACHOLCZYK Philippe

Centre National d'Etudes Spatiales, 18 Avenue Edouard Belin
31055 Toulouse Cedex, France

N94-23846

SUMMARY

SPOT sun-synchronous remote sensing satellites are operated by CNES since February 1986. Today, the SPOT mission and control center (CCM) operates SPOT1, SPOT2 and is ready to operate SPOT3.

During these seven years, the way to operate changed and the CCM, initially designed for the control of one satellite, has been modified and upgraded to support these new operating modes. All these events have shown the performances and the limits of the system.

A new generation of satellite (SPOT4) will continue the remote sensing mission during the second half of the 90's. Its design takes into account the experience of the first generation and supports several improvements.

A new generation of control center (CMP) has been developed and improves the efficiency, quality and reliability of the operations. The CMP is designed for operating two satellites at the same time during launching, in-orbit testing and operating phases. It supports several automatic procedures and improves data retrieval and reporting.

Keywords: SPOT, remote sensing systems, mission and control center, satellite operations, operating organization

1. SPOT SATELLITE MISSION

The decision of SPOT earth remote sensing program was taken by the French government in 1978.

Designed by CNES (Centre National des Etudes Spatiales) and built by France with Belgium and Sweden cooperation, the SPOT system is operational since 1986. It is composed of an image receiving and processing ground segment, a mission and control ground segment and the Spotimage commercial company, which is in charge of earth images selling.

1.1 Primary Mission

The aim of SPOT satellites is to perform images of the whole earth. Equipped by two optical instruments and two magnetic recorders, the satellites can transmit two images of 10 or 20 meters resolution at the same time, in four visible wave bands for SPOT1, SPOT2, SPOT3 and an additional infrared wave band for SPOT4. Each image requests a 25 Mbits per second data link. These images can be recorded on board for further

transmission or directly transmitted down to ground by using a X band (8 GHz) downlink.

The SPOT satellites orbit has been precisely defined to insure the mission requirements:

- it is a polar orbit lightly inclined with regard of the pole axis. This feature enables, with the earth rotation, the whole earth observation.
- the orbit phase is 26 days long and enables a repetitive image taking. As a matter of fact the side angle viewing capability offers stereoscopic images capability and a 5 days long accessibility delay.
- the orbit is roughly circular (830 kms) and the images have the same characteristics whatever the observed point is.
- the orbit is sun-synchronous and implies the overflight of the same point at the same date (1030Z). Then, the images have the same lighting conditions.

These orbit characteristics imply a ground segment rhythm with two visibility groups, one in the morning (about 1030Z) and the other in the evening (about 2230Z).

1.2 Secondary Missions

Since the beginning, SPOT satellites include several passengers which perform secondary missions:

DORIS is a doppler measurement instrument used to determine the satellite orbit with a 10 cm accuracy. DORIS is board on SPOT2, SPOT3 and SPOT4 satellites and has its own control center.

VEGA is a radar responder instrument running independently of the satellite and providing an accurate radar calibration. VEGA is board on SPOT1, SPOT2, SPOT3 and SPOT4.

POAM is a polar ozone atmospheric measurement system which performs a spectral analysis of the polar stratospheric clouds. POAM has its own telemetry link and is monitored by the Consolidated Space Test Center in Sunnyvale, CA. POAM is board on SPOT3.

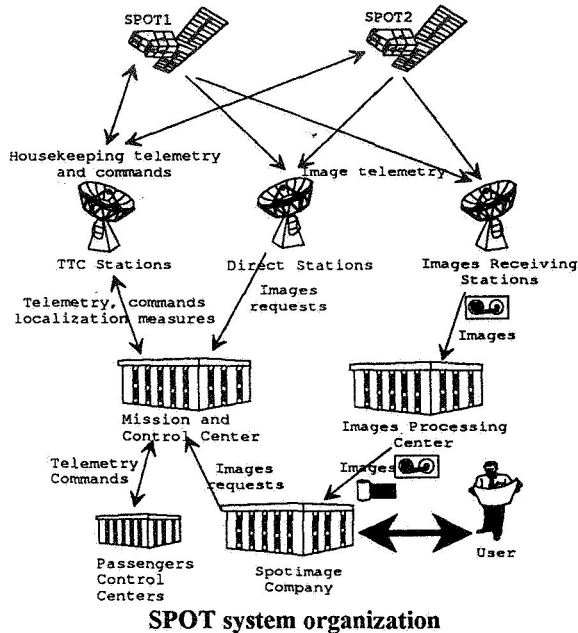
SPOT4 satellite includes some other experiments:

VEGETATION is a remote sensing passenger which can provide 1 km accuracy images used for vegetation monitoring.

PASTEC is a set of seven experiments related to thermal coating degradation, cosmic radiations effects, satellite dynamic characteristics during launch and in-

orbit phases and electrostatic charge measurement.

PASTEL is laser datalink experiment and ESBT is a wide waveband transmitter experiment. They will use a geostationary data relay satellite for payload telemetry link.



SPOT system organization

2. GROUND SEGMENT GENERAL ORGANIZATION

The SPOT ground segment is subdivided into three parts.

The first part is the TTC stations network which enables housekeeping telemetry reception, commands transmission and satellite tracking. The stations are situated near Toulouse in France, Kiruna in Sweden, Pretoria in South Africa, Kourou in Guyana and use the 2 GHz band.

The second part is the image ground segment which includes:

- the image receiving stations which can receive direct or recorded images. These stations are situated near Toulouse and Kiruna.
- the image processing centers which process, archives and produces the satellites images on film or magnetic tape. These centers are settled in Toulouse and Kiruna.
- the direct receiving stations which can only receive direct image telemetry. Today they are twelve spread all over the world.

The third part is the mission and control center (CCM) which is in charge of satellite monitoring and commanding, payload programming, orbit processing and ground facilities coordination.

CCM interfaces with the passengers control centers for passengers monitoring and commanding.

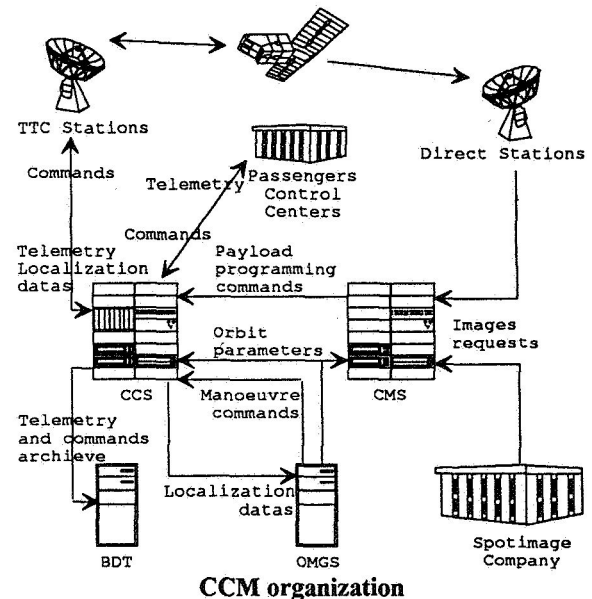
3. CCM ORGANIZATION

Initially designed for the control of one satellite, the SPOT mission and control center has been built between 1980 and 1985. It is subdivided into five parts:

- the control center (CCS) which commands and monitors one satellite.
- the mission center (CMS) which prepares the payload work schedule for the next day for one satellite.
- the orbit determination and manoeuvre center (OMGS) which computes the orbit, prepares the manoeuvres and manages the attitude control for all satellites.
- the technological database (BDS) which archives the whole telemetry of the satellites.
- the satellite simulator (BSI) which can simulate one satellite at a time.

A spare computer can replace either a CCS or a CMS in case of failure of one of the nominal computers.

Two additional computers (CCS and CMS) are used during launch for redundancy and during operating for software and operations validation.



CCM organization

3.1 Control center (CCS)

During a satellite visibility on a 2 GHz TTC station, the CCS is in a real-time mode and performs the following functions:

- real-time monitoring of the telemetry parameters (1400 parameters).
- real-time displaying of 4 semi-graphic parameters synoptics among 40.
- real-time displaying of 12 parameters curves.
- commands transmission to the satellite.
- localization measurements reception from the TTC

station.

Between two visibilities, the CCS is in a postponed mode and performs the following functions:

- preparing the commands plan for the next visibility.
- postponed monitoring of the satellite telemetry: on-board configuration, thermal and power monitoring, trend analysis, etc.

The CCS can also play back the telemetry in case of precise investigation.

The CCS interfaces with passengers control centers for passengers housekeeping telemetry providing and passengers commands transmission.

The software management uses alphanumeric displays.

A CCS can manage one satellite and is built on a SOLAR 16-75 computer.

3.2 Mission center (CMS)

The CMS daily performs automatically the payload work schedule to be send to the CCS by taking into account the requests of the following users.

The first user is Spot Image company which sends the earth parts to be shot requested by the end users. The images situated into a 2500 kms circle around Toulouse or Kiruna are directly taken during a pass, the others are recorded on board for further transmission.

The second users are the direct receiving stations which are spread all over the world. They can book visibilities on their station and send direct image requests until 12 hours before executing.

The third user is the CCM engineers team which can program technological operations on the payload.

Above a latitude of 72 degree, a graphic interface with a map background is used for image programming.

The CMS has also a network node role for exchange between CCS and OMGS.

The software management uses alphanumeric displays.

A CMS can manage one satellite and is built on a SOLAR 16-75 computer.

3.3 Orbit determination and manoeuvre center (OMGS)

The OMGS center calculates twice a day a precise satellite orbit by using the localization measurements, after the morning and evening group of visibilities.

It determines a predicting orbit which is used for the payload programming, the TTC visibilities prediction and the management of on-board attitude events (moon or sun sensors, in orbit position, ...).

It also calculates the connection between on board and UT time.

It calculates and generates the commands to be send for a manoeuvre, in order to insure a right orbit.

It uses graphic displays for orbit evolution curves.

The software management uses alphanumeric displays.

The OMGS center can manage the three satellites and run on a CDC 830 computer.

3.4 Technological database (BDT)

All the data related to the CCM and satellite status are daily archived in the technological database: telemetry, sent commands, orbit parameters, ...

These data can be retrieved and reported on display, paper or plotter.

Today, the BDT manages more than 20 Gigabytes which represents all the status data of CCM and SPOT satellites since their launch.

The software management uses alphanumeric displays.

The BDT can manage the three satellites and run on a CDC 990 computer.

3.5 Satellite simulator (BSI)

The satellite simulator includes a real on-board computer which runs the satellites on-board software. The other equipments and the environment are simulated.

The BSI is used for operations and on-board software releases qualifications.

The BSI can simulate only one satellite at a time and runs on two SOLAR 16-75.

3.6 Human organization

The human organization is subdivided into several parts:

- the center chief with a secretary
- the satellite management service (CB) which is responsible for satellites control and operations, simulator management (9 people).
- the ground technical support service (STS) which is responsible for ground hardware and software management (11 people).
- the operational support service (OPS) which is responsible for ground facilities operation and includes controllers (30 people).

One CB, CCS, CMS and OMGS engineer is always available in case of ground or on-board anomaly.

3.7 Software and documentation

CCM software has been developed in Fortran and contains more than 1.3 millions of code lines.

In addition of development documentation, CCM team uses flight plan and ground operations plan documents. These documents describe all the operating phases (launch, in-orbit test and operating), represent more than 4,000 pages and have been produced with word processing software.

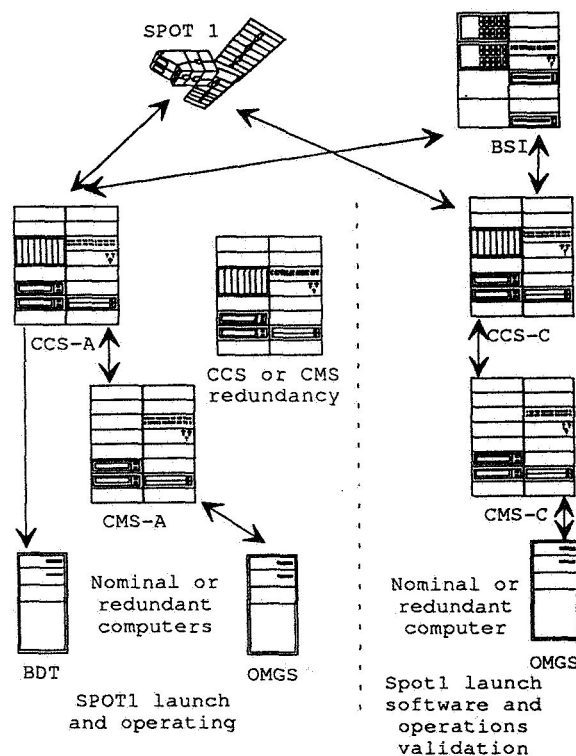
4. CCM SUCCESSIVE ORGANIZATIONS

During these seven years of operation, the CCM organization has been changing several times with respect to the great events of SPOT system.

4.1 Launching and operating one satellite

The first satellite, SPOT1, was launched on February 21st 1986. The CCM has been set up one year before in order to perform technical and operational qualification.

The CCM was managed from 0500Z to 0200Z by three controllers.



One satellite organization

4.2 Operating two satellites

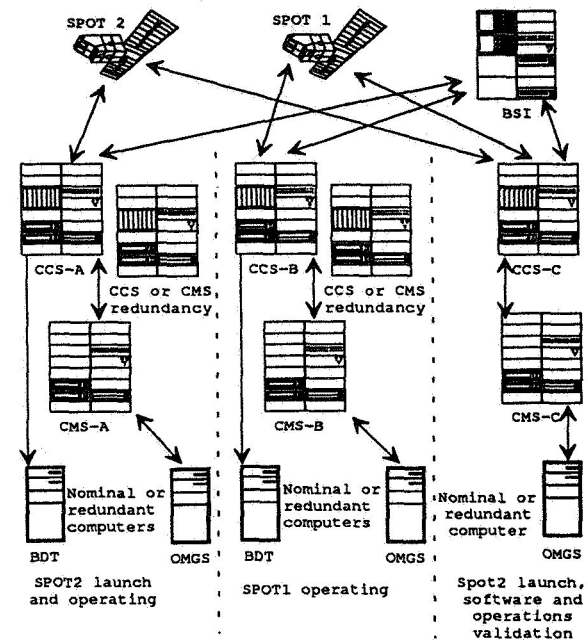
The second satellite, SPOT2, was launched on January 21st 1990. The technical and operational qualification started one year before.

To achieve the new mission, the computers which can manage only one satellite have been duplicated: CCS, CMS and the spare computer. The software had been designed to be parametrized by satellite.

Both satellites were operated together for image taking until the end of 1990. Both CMS were interconnected in order to share the images request pool.

Automatization of the postponed satellite monitoring management on CCS has been upgraded in order to decrease the controller workload.

Each CCM (one per satellite) was managed from 0500Z to 0200Z by two controllers.



Two satellites organization

4.3 Operating one satellite and housekeeping the other

From January 1991 until March 1992, there were no payload activity on SPOT1 satellite. It was controlled in a backup mode in order to keep it safe and ready to take again images if needed.

The CCM organization was the same with a reduce activity of SPOT1 CMS, which had only a network node role. There were only two visibilities in the morning and the evening.

SPOT2 CCM was managed from 0500Z to 0200Z by two controllers.

SPOT1 CCM was managed from 0600Z to 1430Z and 1830Z to 0200Z by one controller.

During this period, CCM made some technological operations on SPOT1 in order to determine the in-orbit performances of the satellite:

- earth and sun sensors inhibition, which shown a very good attitude control with the gyroscopes only: less than 0.3 degree of drift after 24 hours.

- magnetic couplers inhibition, which are used for momentum wheels unsaturation: highlighting of an unforeseen wide magnetic moment without impact on attitude control.

- stop of the solar array spin: when the solar array is perpendicular and parallel to the speed: there is a rate of two between the aerodynamic torques.

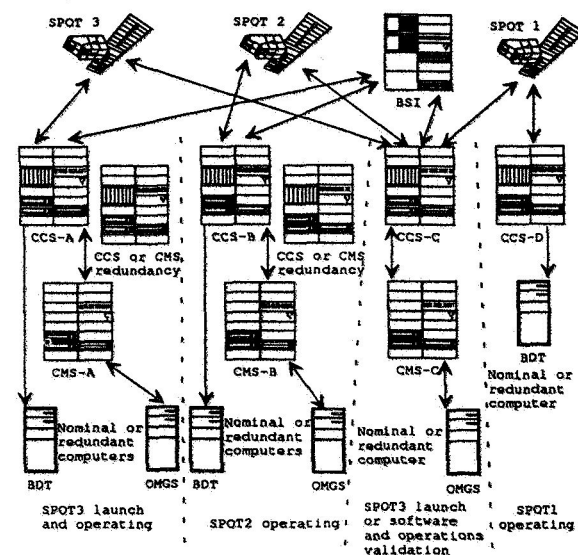
All these operations have shown a great robustness of the satellite attitude control, which is widely inside the requirements, and confirmed the magnetic and aerodynamic models used.

4.4 Operating one satellite, housekeeping the second and launching the third

SPOT3 will be launched in September 1993 and CCM will be upgraded in order to store SPOT1 and operate SPOT2 during SPOT3 ground qualification and launch.

SPOT1 mission will stop at the beginning of November 1993. The modified on-board software will insure attitude control of the satellite without ground corrections.

Satellite status will only be checked during two visibilities in the morning and the technological database will continue to be filled.



Three satellites organization

SPOT2 and SPOT3 CCM will be managed from 0500Z to 0200Z by four controllers.

SPOT1 CCM will only be managed during the two check visibilities by one of the four controllers.

4.5 Results and conclusions

The CCM organization settled by CNES for SPOT satellites operating is efficient, reliable and flexible.

Indeed SPOT1, designed for 2 years lifetime, is always operational and the CCM, designed in 80's for one

satellite, is today able to operate 3 satellites.

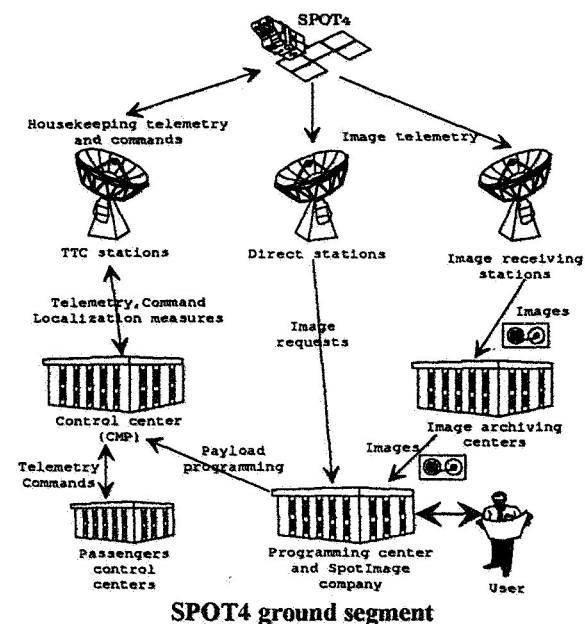
Finally satellites control has never been interrupted and there were no mission interruption because of ground failure.

5. SPOT4 NEW GENERATION SATELLITE

SPOT4 satellite will be launched between 1995 and 1998. Because of a larger telemetry (more than 6,000 parameters in the database) and a more flexible on-board software, this new generation satellite provides better subsystems observing and commanding.

SPOT5 satellite is today on study and will continue the remote sensing mission.

5.1 Ground segment organization



SPOT4 ground segment

SPOT4 ground segment has been designed to support two satellites at a time and is subdivided into three parts.

The first part is the 2 GHz TTC stations network which includes Toulouse, Kiruna, Kourou stations and a new one on Kerguelen island.

The second part is the image ground segment which includes:

- the same image receiving stations as SPOT1.
- the same direct receiving stations.
- Kiruna and Toulouse image archiving centers.

As regards to SPOT1, the image processing centers and the payload programming part of the mission center are settled at the premises of the Spotimage company.

The third part is the control center (CMP) which is in charge of satellite monitoring and commanding, orbit processing and ground facilities coordination.

Today, the CMP is on the way of integration at CNES in Toulouse and its technical qualification will begin in 1994.

5.2 CMP organization

CMP center has been designed for two satellites and is subdivided into four parts (Ref. 1):

- the control center (CCS) which commands and monitors one satellite.
- the management center (CGS) which prepares the payload command plan, computes the orbit, prepares the manoeuvres, archives the whole telemetry and interfaces with the passengers control centers.
- the video server, used during launch phase, which broadcast up to sixteen telemetry synoptics on graphic displays.
- the satellite simulator (BSSO) which can simulate one satellite at a time.

Using of up-to-date hardware and software enables functional improvements which improve operations efficiency, quality and reliability.

Two satellites upgrading shall be simply done by adding some new CCS hardware.

5.2.1 CMP improvements

Controllers or engineers use a window based interface on colour graphic displays for satellite or CMP management.

The daily CMP management can be done either manually or automatically (Ref. 2).

At the beginning of the day, the controller will prepare the CMP day workschedule on the CGS and commands preparing, telemetry monitoring and commands sending during a visibility, postponed telemetry monitoring, telemetry archiving will run automatically. He monitors the CMP activities on a timeline graphic chart and acts only in case of anomaly.

Telemetry and commands database management has been improved by using Oracle database software and interfacing all CMP functions with database extracts.

Technological database extracts can be graphically performed and printed on CGS displays.

Engineers PC's will be connected by a network to the CGS in order to import technological database extract or configuration files and perform them on PC's softwares.

5.2.2 Human organization

One controller can manage two satellites at a time and both control and management centers associated.

The CMP will be managed from 0600Z to 2200Z by

two controllers.

The human organization will be subdivided into several parts:

- the CMP chief with his assistant and secretary.
- the satellite management team which is responsible for satellite control and operations, simulator management (5 people, 1 always on duty).
- the ground support team which is responsible for ground hardware and software management (4 people, 1 always on duty).
- the operations team which is responsible for ground facilities operating and includes controllers (6 people).

5.2.3 Documentation

A specific software will provide tools for operation plans preparation.

Flight and ground operation plans will be merged and described at the same time on PROCES tool, by using a specific language and by referencing either board or ground actions: used synoptics, sent commands, checked telemetry parameters, ground processing workschedule, operator acting.

Operation plans coherence with the data existing on the CMP is automatically checked before printing. PROCES can also simulate procedure execution and check dynamical coherence.

PROCES tool will improve efficiency and reliability of the satellite operations.

5. CONCLUSION

Today, with the CCM, CNES has a 7 year experience on SPOT remote sensing satellites operating and will operate 3 satellites at the same time during 1993.

This experience has been used for the design of the future remote sensing programs and CNES prepares the CMP, a new generation operating center, which will improve efficiency, quality and reliability of the operations.

6. REFERENCES

1. Zaouche, G. 1992. SPOT4 operational control center. In SpaceOps 92.
2. Fratter, I. 1992. Agenda: a task organizer and scheduler. In SpaceOps 92.

SESSION B

SYSTEM-LEVEL ARCHITECTURES

CO-CHAIRPERSONS

J.-F. Kaufeler, ESA European Space Operations Centre, Germany
W. S. Tai, JPL, California Institute of Technology, USA

Summary	89
J.-F. Kaufeler	
B.1 A Reference Model for Space Data System Interconnection Services	91
J. Pietras, MITRE Corporation, Greenbelt, Maryland, USA; G. Theis, ESA European Space Operations Centre, Darmstadt, Germany	
B.2 An OSI Architecture for the Deep Space Network	97
W. R. Heuser and L. P. Cooper, JPL, California Institute of Technology, Pasadena, California, USA	
B.3 ATOS-1: Designing the Infrastructure for an Advanced Spacecraft Operations System	103
K. J. Poulter and H. N. Smith, Logica Space and Communications Ltd., London, UK	
B.4 Distributed Science Operations for JPL Planetary Missions	113
R. D. Benson and P. B. Kahn, JPL, California Institute of Technology, Pasadena, California, USA	
B.5 SCOS-II: ESA's New Generation of Mission Control Systems	119
J.-F. Kaufeler and N. C. Head, ESA European Space Operations Centre, Darmstadt, Germany	
B.6 Operations Concepts for Mars Missions with Multiple Mobile Spacecraft	125
W. C. Dias, JPL, California Institute of Technology, Pasadena, California, USA	
B.7 Preparing the DSN Operations Concepts	131
A. J. Salazar, M. D. Bailey, N. R. Kuo, and B. M. Wilkinson, JPL, California Institute of Technology, Pasadena, California, USA	
B.8 NASDA Satellite Mission Operation System and Operations	137
K. Yamaya, F. Yoshida, H. Noguchi, T. Takahashi, and K. Tanaka, NASDA, Tsukuba, Japan	
B.9 NASA/Goddard Space Flight Center's Testbed for CCSDS Compatible Systems	143
R. D. Carper, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA	
B.10 SPOT4 Operational Control Center (CMP)	149
G. Zaouche, CNES Toulouse Space Centre, France	

SUMMARY — SESSION B

J.-F. KAUFELER

European Space Operations Centre

This session was particularly well attended. Great interest was evident on the part of the audience due to the very high quality of the presentations and professional answers given to the questions.

System architecture is always a key session in this kind of symposium, since the topic addresses the core infrastructure on which a space organisation bases its ground segment. In this respect, this session successfully met expectations. A number of fundamental technical observations were made and functional trends noted; moreover, some visions for future work were formulated.

TECHNICAL OBSERVATIONS

It has to be noted that CCSDS standards have been largely introduced in the new ground system architectures — in the form of prototypes/testbeds or even in operational systems (B.1, B.5, B.9). This enables mission data (telemetry and telecommand) to be more flexible than before (e.g., data units rather than time-division multiplexed data) and makes it possible to exchange data between space/ground and ground/ground in a cross-support configuration.

The general architecture of those systems seems to converge towards similar concepts based on open and distributed architectures. The reason is probably due to the wide appearance of standards and corresponding commercial products (e.g., ISO/OSI, OSF) on the market. Clearly, the trend is towards C++-based and UNIX-based systems (B.2, B.4, B.10). Consequently, these architectures follow a very similar layered model; certainly the CCSDS standards have played a major role in this direction.

In the last two years, considerable work has been devoted towards exploiting knowledge-based and artificial intelligence techniques. However, it has to be noted that this requires a non-negligible promotion effort (towards the future users) and those developments are still at a prototype level of self-standing complementary systems like mission planning (B.3).

FUNCTIONAL TRENDS

Because of the factors stated above, most mission-control systems are implemented in a

generic manner for reuse in different missions. Also, the notion of multimission systems is developing in anticipation of missions composed of a series of small satellites (B.6).

It appears that ground networks (in particular the ground stations) will follow a similar operations concept as the mission itself (B.7, B.8) in the future and will be based on similar functional systems that will eventually be based on reusable software components.

VISIONS FOR THE FUTURE

Although SPACEOPS 92 was primarily dedicated to ground segment infrastructure aspects, it is a fact that the advent of new operations concepts based on new system architectures has a direct impact on onboard architectures.

It has to be noted that the ground segment community is progressing faster than the space segment community in terms of software technology. The reason for this is that one is much less constrained on the ground than on board a spacecraft (e.g., memory resources, environmental considerations, etc.). Therefore, the space community is somehow reluctant to embark on implementing these concepts. Furthermore, these two communities are often working in isolation. There is certainly a need for increased collaboration between space engineers and ground engineers in the concept phase of a mission, which has to be seen as a complete system. Also, an increased effort in the promotion of ground segment concepts towards projects is required.

Open architectures are characterised by their growth potential (new functions, new applications) at given system-layer interfaces. More emphasis should be devoted to identifying and describing these interfaces in order to enable better reusability of those systems for cross-support purposes, easy introduction of mission-specific functions, and eventually for marketing purposes.

Since the allotted time for presentations was too short to address this subject, it may be worthwhile to consider a separate session on system interfaces for the next SPACEOPS symposium. This would provide an opportunity for participants to be made aware of interface possibilities, and this could lead to more effective system collaborations.

A REFERENCE MODEL FOR SPACE DATA SYSTEM INTERCONNECTION SERVICES

N94-23847

John Pietras

Gerhard Theis

The MITRE Corporation
Greenbelt, Maryland, USA

ESA/ESOC
Darmstadt, Germany

ABSTRACT

The widespread adoption of standard packet-based data communication protocols and services for spaceflight missions provides the foundation for other standard space data handling services. These space data handling services can be defined as increasingly sophisticated processing of data or information received from lower-level services, using a layering approach made famous in the International Organization for Standardization (ISO) Open System Interconnection Reference Model (OSI-RM). The Space Data System Interconnection Reference Model (SDSI-RM) incorporates the conventions of the OSI-RM to provide a framework within which a complete set of space data handling services can be defined. The use of the SDSI-RM is illustrated through its application to data handling services and protocols that have been defined by, or are under consideration by, the Consultative Committee for Space Data Systems (CCSDS).

Key Words: Space communications, CCSDS, telemetry, telecommand, ISO/OSI

1. INTRODUCTION

Over the past decade, the world community of civilian space agencies has fostered the evolutionary adoption of standards for operational space data communication systems. Early efforts at standardization concentrated on the transfer of data across the space-ground radio frequency (RF) link. These efforts resulted in a packet-based statistical multiplexing alternative to time division multiplexing (TDM); powerful forward error correction for telemetry; and a reliable command delivery protocol. These standard mechanisms are documented in the CCSDS Recommendations for Packet Telemetry (Ref. 1) and Telecommand (Refs. 2 - 4).

Prompted by the prospect of an international space station containing hundreds of instruments and platform control functions communicating with a world-wide user community on the ground, CCSDS extended the scope of standardization. The expanded-scope services and protocols, documented in the CCSDS Recommendation for Advanced Orbiting Systems (AOS) (Ref. 5), comprise (a) an

expanded suite of space link services and protocols (e.g., bitstream, isochronous insert), and (b) end-to-end data transfer services (e.g., onboard instrument to ground-based telescope workstation). The AOS Recommendation introduces the concept of the CCSDS Principal Network (CPN), which is the concatenation of onboard, space link, and ground subnetworks.

The aforementioned CCSDS Recommendations were developed by starting at the space-ground link and working out. This "RF link out" approach is in contrast to that taken in the development of Open System Interconnection (OSI), an initiative of the International Organization for Standardization (ISO) which was to create a suite of services and protocols that would allow application programs on any two arbitrary end systems (aka nodes) to communicate. The OSI services and protocols have been created on the framework of the OSI Reference Model (OSI-RM) (Ref. 6). The OSI-RM allocates the functions of data communication to the now-famous seven layers: physical, data link, network, transport, session, presentation, and application. Because the OSI-RM addresses the complete range of functions that must be performed to allow two applications to communicate, the developers of the specific services and protocols at each layer were able to identify and mitigate overlaps and shortcomings within the suite of services/protocols.

Such a model for the interconnection of application processes is not limited to OSI services and protocols. In the context of space data systems, such a model could be used to:

- Provide a framework within which all aspects of space data interchange can be identified
- Identify which aspects of space data interchange are suitable to standardization
- Consolidate fundamental concepts for analysis of the data and information processing requirements of space missions
- Foster a terminology common to the space data and information handling community
- Provide the overall context that allows the standardization effort to proceed in multiple "narrower" activities better suited to the time and resource limitations of the standardization process

PRECEDING PAGE BLANK NOT FILMED

Highlight data and information handling requirements common to space-based and ground-based systems, and thus foster the application of approaches and technologies common to both.

This paper introduces such a model, called the Space Data System Interconnection Reference Model (SDSI-RM). The scope of the SDSI-RM encompasses all communications between user applications exchanging space-related data.

The SDSI-RM is a synthesis of several recent space data system modeling activities: the CCSDS ground infrastructure reference model used in the development of CCSDS-standard ground infrastructure services (Ref. 7); a reference model for the European Space Agency (ESA) Space Data Network developed by Theis (Ref. 8); and a space mission interconnection model being developed for cross-support services between the Goddard Space Flight Center and the Jet Propulsion Laboratory (Ref. 9). These recent modeling efforts focus on different subsets of the overall space data communication problem, and do not address certain communication profiles (such as inter-spacecraft communications) and classes of service (such as end-to-end protocols and services). The goal of the SDSI-RM is to abstract the essential features common to these other models, and to provide the conceptual framework for defining the currently missing profiles and classes.

Owing to the introductory nature and brevity of this paper, the SDSI-RM as presented here glosses over a variety of rough edges that surface through detailed analysis. The intent here is to present the core features of the SDSI-RM.

2. SDSI-RM CONCEPTS AND TERMINOLOGY

The SDSI-RM adopts concepts and terminology of the OSI-RM, and augments those as necessary.

At the highest level, the SDSI-RM is partitioned into *architectural profiles*. These profiles correspond to major scenarios for the interconnection of space data-handling systems. Each architectural profile is defined in terms of the telecommunication services that interconnect two *user applications*.

User applications are part of, or attached to, *end systems*. Through the cooperation of *service entities* on each of the end systems, a telecommunication service is provided between the user applications. The telecommunication services are provided in a layered manner: a service entity in one end system exchanges service data and protocol information with its peer service entity in the other end system by using the service offered by the layer below. The

service entity at a given layer is simultaneously the *provider* of a service to the layer above, and a *user* of the service provided by the layer below, with each successive service layer adding value to the service provided by the layer below. This layering approach is successively applied from the medium that physically connects the end systems up to the user applications.

The case where two end systems are physically connected by a single physical medium is a special one. In the more general case, one or more *intermediate systems* lie between the end systems. An intermediate system is used to interconnect across multiple subnetworks, and/or when a service/protocol conversion is necessary between end systems.

The services within each architectural profile are grouped into *service classes*. Service classes are groupings of functionally-related services. A service class may be decomposed into more-specific, more narrowly defined subclasses. At the bottom of the subclassing hierarchy are concrete services (e.g., ISO Connectionless Transport Service (Refs. 10 & 11) using Transport Protocol class 4 (Ref. 12)). In most cases such concrete services have not yet been specified. For those cases, the SDSI-RM provides a place holder for requirements against to-be-specified services and protocols.

The specification of the different service classes within an architectural profile is based primarily on layer functionality vis-a-vis the OSI seven-layer model. However, classification by OSI layer alone does not address the different contexts in which those services are used. For example, the same transport service may be used to connect two end systems directly (an end-to-end context), and also be used to connect an end system to an intermediate system (a subnetwork context). The SDSI-RM classes reflect these contexts, and it is possible for the same services to appear in several service classes with the same architectural profile.

There are four architectural profiles in the SDSI-RM: space-ground, inter-spacecraft, terrestrial, and intra-spacecraft. The space-ground architectural profile is the focus of most modeling efforts, so it will be presented first and in the greatest detail.

3 SPACE-GROUND ARCHITECTURAL PROFILE

The space-ground architectural profile deals with interactions between space-borne and ground-based user applications. The space-borne user applications may be sensors, platform control systems, or, in the case of human-occupied spacecraft, astronauts or cosmonauts. The ground-based user applications that

communicate with the space-borne applications include science data handling systems, telescientists, platform control center processes, and ground controllers.

The space-ground architectural profile contains seven top-level service classes:

- CPN Communication services class
- CPN Transport services class
- CPN Internet services class
- Space Link Access services class
- Space Link services class
- Terrestrial Telecommunication services class
- Spacecraft Telecommunication services class

Figure 1 illustrates the space-ground architectural profile, where the service classes are represented as boxed groupings of corresponding service elements.

3.1 CPN Communication Services Class

The CPN Communication services class provides the interconnected user applications with services associated with the OSI-RM's session, presentation, and application layers. These services are realized through the interaction of peer entities directly associated with the communicating user applications. The CPN Communication services class contains three subclasses: the ISO/OSI Communication services class, the Space Operations services class, and the Space Information Interchange services class.

The ISO/OSI Communication services class consists of a set of specific ISO-standard services and protocols at the application, presentation, and session layers. This suite is available for flight projects with requirements for OSI compliance. The ISO/OSI Communication services use the ISO/OSI Transport service class of the CPN Transport services class (see 3.2).

While the ISO/OSI communication services class provides many services of potential use in space flight operations, a growing body of evidence shows that use of the ISO/OSI protocols across a space-ground link may provide unacceptable performance in many operational scenarios. The Space Operations Communication services class is intended to provide a set of information transfer services equivalent to those of the ISO/OSI Communication services class, but modified for optimal performance in the space operations environment. The Space Operations Communication services class does not now exist, but CCSDS is contemplating a program of work to develop such a suite. Candidates for early development are a space-optimized file transfer service/protocol and commanding service/protocol. The Space Operations Communications services use

the Space Operations Transport service of the CPN Transport services class (see 3.2).

The Space Information Interchange services class comprises application- and presentation-layer services constructed around the CCSDS Standard Formatted Data Unit (SFDU). The SFDU is the currency of a standard system for encapsulating information in a way that enhances the identifiability, locatability, and archivability of that information. The Space Information Interchange services support the creation, cataloging, packaging, locating (finding), browsing, evaluation, and retrieval of SFDUs.

The Space Information Interchange services are currently being defined within CCSDS. As currently modeled in the SDSI-RM, the Space Information Interchange services can use the services of either the ISO/OSI or Space Operations Communication services class (or both) to connect across the space-ground link.

3.2 CPN Transport Services Class

The CPN Transport services class transfers data across the multiple subnetworks between the space-based and ground-based application processes, while providing the level of reliability that is required by the communicating user applications. The CPN Transport services class contains two subclasses, the ISO/OSI Transport service (Refs. 10 & 11) and the Space Operations Transport service.

The ISO/OSI Transport service is available for flight programs requiring OSI compliance. The ISO Transport service uses the CPN Internet service (see 3.4).

The Space Operations Transport service is a transport service (with supporting protocol) tailored to the space-ground telecommunication environment. The Space Operations Transport service and corresponding protocol do not now exist, but CCSDS will begin a program of work to develop them in the near future. The current thinking is that the Space Operations Transport service will use the CCSDS Path service of the Space Link Access services class (see 3.3) to provide an internetwork connection between the space-based and ground-based end systems.

3.3 Space Link Access Services Class

The Space Link Access (SLA) services class allows application processes that are remote from the termination of the space link to access the Space Link Services (see 3.5) that are provided at that link

termination. The SLA services provide access to Space Link service data in real-time and on a delayed-delivery basis.

The model of SLA services described in this paper actually addresses the simple case of providing SLA services when all of the Space Link service protocol processing is performed at the space link terminal. A general model of the SLA services exists which incorporates distribution of Space Link service protocol processing. The simple-case model described here is a subset of that general model.

Within the space-ground architectural profile, the SLA services class contains two subclasses, the Ground-based SLA (GSLA) services class and Space-based SLA (SSLA) classes. While the SSLA and GSLA services classes provide conceptually equivalent services, they differ in the specific characteristics of those services and the protocols used to provide those services. These differences arise from differences in the environments: the closed, local-area-networked, single-management-domain environment of the spacecraft vs. the relatively open, distributed, wide-area-networked, multiple-management-domain environment of the ground systems.

All but one of the SLA services are always accessed directly by the user application, without the use of intermediate CPN Communication or Transport services. These directly-accessed SLA services provide several different ways for moving mission-unique-format data between the space-based and ground-based user applications. The Path service (Ref. 5) does not always bypass the intervening layers: it provides a space-operations-optimized network service through the space link in support of Space Operations Transport and Space Operations Communication services. (The intervening layers may also be bypassed for Path service, so that the user applications may operate mission-unique communications and transport protocols through the Path service.)

The SLA services are provided by a client-server mechanism, with the server entity accessing Space Link Services in an intermediate system that terminates one end of the space link, and the client entity contained within the user application's end system. The GSLA server and client entities interconnect via Terrestrial Telecommunication services (see 3.6), and the SSLA server and client entities interconnect via Spacecraft Telecommunication services (see 3.7).

Standard GSLA services are currently under active development within CCSDS. The SSLA class is not being considered for multi-mission standardization at

this time. However, the SSLA class provides a reference model for individual flight programs to use in developing onboard service architectures that complement the services that will be encountered on the ground.

3.4 CPN Internet Service Class

The CPN Internet service class contains one concrete service, the CPN Internet service (Ref. 5), which provides the ISO Connectionless Network Service (Refs. 13 & 14) between the space-based and ground-based end systems. CPN Internet service entities exist in the two end systems, and in the intermediate systems containing the Space Link service entities.

3.5 Space Link Services Class

The Space Link services class supports the SLA services class by establishing and maintaining a data link between the ground infrastructure and the space infrastructure onboard a spacecraft. Collectively, the services in the Space Link services class:

- Establish and maintain the RF link between the ground and the mission spacecraft.
- Perform the protocol processing necessary to transfer various CCSDS-defined Space Link service data units across the space link.

The Space Link services are the services corresponding to the CCSDS Recommendations for AOS, Telecommand, and Packet Telemetry.

3.6 Terrestrial Telecommunication Services Class

In the context of the space-ground architectural profile, the Terrestrial Telecommunication services class provides telecommunication connectivity between the server and client entities of the GSLA services. The Terrestrial Telecommunication services class contains two subclasses, the Terrestrial Communication services class and the Terrestrial Transmission services class.

The Terrestrial Communication services class provides ISO/OSI services and protocols at the application, presentation, and session layers. The applicability of specific services within this class is under study. For example, File Transfer, Access, and Management (FTAM) (Ref. 15) is a candidate for use in transferring time-delayed files of space data between GSLA client and server.

The Terrestrial Transmission services class provides ISO/OSI services and protocols at the transport, network, data link, and physical layers. These services are used to support the Terrestrial

Communication services, and to provide ISO Transport and Network services for the transfer of real-time space data between GSLA client and server.

Selection of specific ISO/OSI services and protocols to support the GSLA services is a future item of work.

3.7 Spacecraft Telecommunication Services Class

In the context of the space-ground architectural profile, the Spacecraft Telecommunication services class provides telecommunication connectivity between the server and client entities of the SSLA services. The Spacecraft Telecommunication services class contains two subclasses, the Spacecraft Communication services class and the Spacecraft Transmission services class.

Conceptually, the Spacecraft Telecommunication services class provides ISO/OSI services and protocols offered in the Terrestrial Telecommunications services class, but specific services may differ from those selected for the ground network because of different requirements to support the SSLA services.

There is currently no work planned within CCSDS to standardize Spacecraft Telecommunication services or protocols.

4. INTER-SPACECRAFT ARCHITECTURAL PROFILE

The inter-spacecraft architectural profile deals with the communication between application-bearing spacecraft. An example is the communication between a free-flyer and Space Station or Shuttle during proximity operations. This profile is a simple variant of the space-ground profile, with another "space half" substituted for the ground half of the space-ground profile.

CCSDS is not currently working on this architectural profile.

5. TERRESTRIAL ARCHITECTURAL PROFILE

The terrestrial architectural profile deals with the communication of space-related data between two applications on the ground. This profile essentially consists of the Terrestrial Telecommunication services class found in the space-ground profile, augmented by the Space Information Interchange services class. The assumption that the terrestrial profile is populated by standard ISO/OSI protocols and services is basically a default position in lieu of any evidence that these standard services are

inadequate to the requirements. CCSDS is not currently working on this architectural profile.

6. INTRA-SPACECRAFT ARCHITECTURAL PROFILE

The intra-spacecraft architectural profile deals with communication between two applications onboard the same spacecraft. This profile essentially consists of the Spacecraft Telecommunication services class found in the space-ground profile, augmented by the Space Information Interchange services class. CCSDS is not currently working on this architectural profile.

7. REFERENCES

1. CCSDS. 1987. *Recommendation for Space Data System Standards. Packet Telemetry*. CCSDS 102.0-B-2, CCSDS Secretariat, NASA, Washington, DC.
2. CCSDS. 1987. *Recommendation for Space Data System Standards. Telecommand, Part 1 - Channel Service: Architectural Specification*. CCSDS 201.0-B-1, CCSDS Secretariat, NASA, Washington, DC.
3. CCSDS. 1987. *Space Data System Standards. Telecommand, Part 2 - Data Routing Service: Architectural Specification*. CCSDS 202.0-B-1, CCSDS Secretariat, NASA, Washington, DC.
4. CCSDS. 1987. *Recommendation for Space Data System Standards. Telecommand, Part 3 - Data Management Service: Architectural Specification*. CCSDS 203.0-B-1, CCSDS Secretariat, NASA Washington, DC.
5. CCSDS. 1989. *Recommendation for Space Data System Standards. Advanced Orbiting Systems, Network and Data Links: Architectural Specification*. CCSDS 701.0-B-1, CCSDS Secretariat, NASA, Washington, DC.
6. ISO/IEC. 1984. *Information Processing Systems - Open Systems Interconnection: Basic Reference Model*, International Organization for Standardization and International Electrotechnical Committee, International Standard 7498.
7. CCSDS. 1992. *Recommendation for Space Data System Standards. Description of CCSDS Ground Infrastructure Cross-Support Services*, CCSDS, draft White Book.

8. Theis, Gerhard. 1992. *A Reference Model for the Space Data Network*, ESA/ESOC, Darmstadt, Germany.

9. NASA. 1992. *NASA Concept Presentation: Space Mission Interconnection Model and Service Description*, presented at the CCSDS Panel 3 meeting of November 1992 ant the Goddard Space Flight Center, Greenbelt, MD.

10. ISO/IEC. 1986. *Information Processing Systems - Open Systems Interconnection - Transport Service Definition*. International Organization for Standardization and International Electrotechnical Committee, International Standard 8072.

11. ISO/IEC. 1986. *Information Processing Systems - Open Systems Interconnection - Transport Service Definition - Addendum 1: Connectionless Mode Transmission*. International Standard 8072/AD 1.

12. ISO/IEC. 1986. *Information Processing Systems - Open Systems Interconnection - Transport Protocol Specification*. International Standard 8073.

13. ISO/IEC. 1987. *Information Processing Systems - Data Communications - Network Service Definition*. International Standard 8348.

14. ISO/IEC. 1987. *Information Processing Systems - Data Communications - Network Service Definition - Addendum 1: Connectionless-mode Transmission*. International Standard 8348/ AD 1.

15. ISO/IEC. 1990. *Information Processing Systems - File Transfer, Access, and Management*. International Standard 8571.

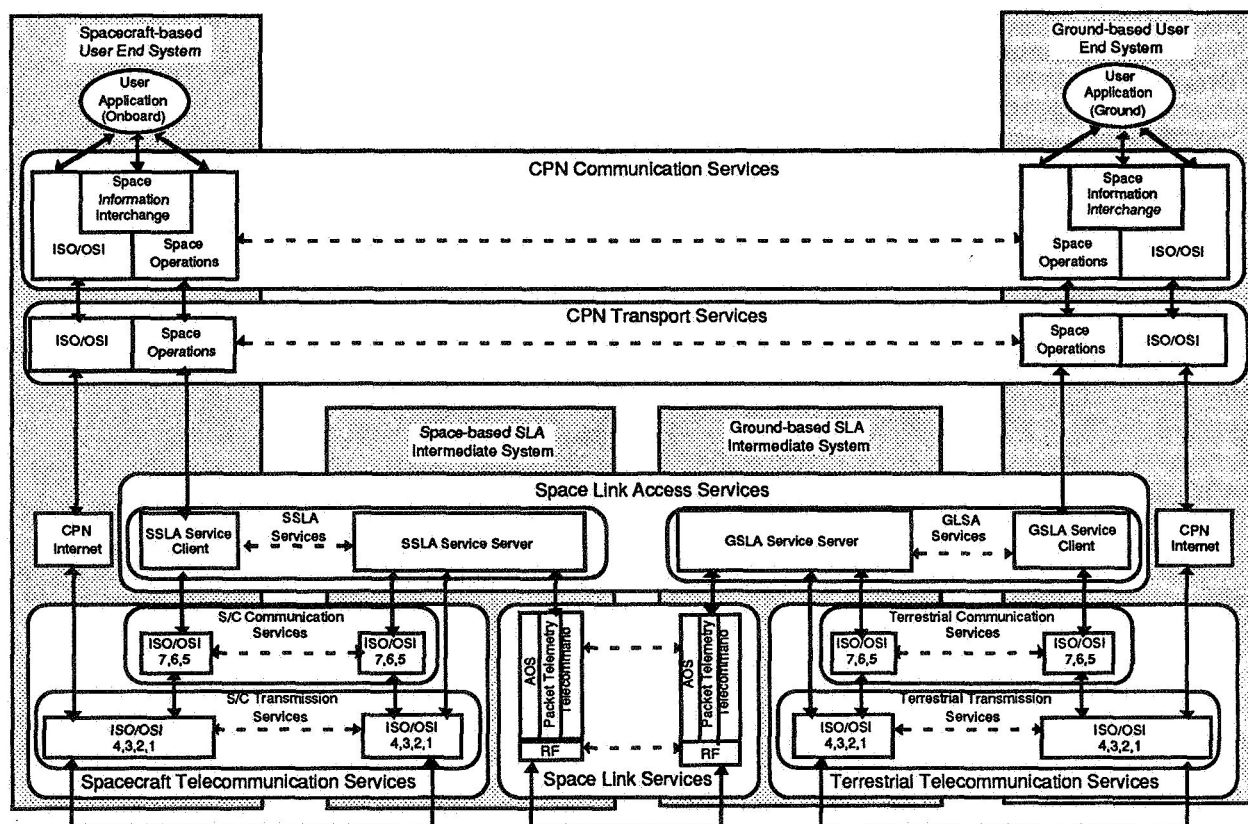


Figure 1. Space-Ground Architectural Profile

AN OSI ARCHITECTURE FOR THE DEEP SPACE NETWORK

W. Randy Heuser
 Lynne P. Cooper
 Monitor & Control Technology Group
 Jet Propulsion Laboratory
 California Institute of Technology
 4800 Oak Grove Drive
 Pasadena, CA 91109

N94-23848

ABSTRACT

The flexibility and robustness of a monitor and control system are a direct result of the underlying inter-processor communications architecture. A new architecture for monitor & Control at the Deep Space Network Communications Complexes has been developed based on the Open System Interconnection (OSI) standards. The suitability of OSI standards for DSN M&C has been proven in the laboratory. The laboratory success has resulted in choosing an OSI-based architecture for DSS-13 M&C. DSS-13 is the DSN experimental station and is not part of the "operational" DSN; it's role is to provide an environment to test new communications concepts can be tested and conduct unique science experiments. Therefore, DSS-13 must be robust enough to support operational activities, while also being flexible enough to enable experimentation. This paper describes the M&C architecture developed for DSS-13 and the results from system and operational testing.

Keywords: OSI, Monitor & Control, GOSIP, Deep Space Network

1. INTRODUCTION

The flexibility and robustness of a monitor and control system are a direct result of the underlying inter-processor communications architecture. The Monitor and Control (M&C) system for the Deep Space Network Communications Complexes (DSCC) is based on an architecture developed 10 years ago -- before the advent of the international standards that exist today. As a result, the existing M&C architecture is difficult to operate, maintain, and update.

The emergence of the Open Systems Interconnection (OSI) standards and the concept of an open systems architecture hold great promise for providing improved operational environments. The application of OSI-based commercial products to support inter-processor communications is expected to improved inter-operability while resulting in substantial cost savings. The Government Open Systems

Interconnection Profile (GOSIP), a mandatory Federal Information Processing Standard (FIPS) for all new computer systems, was adopted to provide computer security for all government systems and to promote the development of OSI technology in the commercial arena.

The problems that face the DSN are assessing the applicability of the OSI standards for DSN systems and identifying migration paths from the existing systems to OSI-based systems. To determine whether the OSI standards were capable of meeting the needs of the DSN, an OSI testbed was developed to investigate the process control standards (MMS/MAP). The testbed consisted of three elements: the Very Long Baseline Interferometry (VLBI) subsystem, the power meter, and the monitor & control computer. The goal of the testbed was to establish client-server relationships between M&C and the two subsystems across a LAN, to create a client-server relationship between the two subsystems, and to assess performance. The testbed was highly successful and the results indicated that an OSI-based architecture was feasible for DSN Station M&C [Ref. 1].

The success of the OSI-based approach resulted in its selection as the M&C architecture for DSS-13 -- an experimental Deep Space Station in the DSN. DSS-13 is not part of the regular operational DSN. Rather, it serves two purposes: as a test-bed for new concepts in deep space communications, and as a resource for performing unique science experiments. DSS-13 consists of a new 34-Meter Beam Wave Guide Antenna, two advanced receivers, Weather Station, Total Power Radiometer, Microwave Switch Controller, VLBI Wide Channel Band System, Water Vapor Radiometer, Data Handler, and the Monitor & Control System. The goal for 1992 is to develop, install, and test an operational M&C System to support DSS-13 operations and experimentation using this core set of equipment. This paper will describe the architecture of the DSS-13 M&C system, discuss how the OSI standards were applied, and provide results from systems and operational testing of the M&C system.

2. DSS-13 MONITOR & CONTROL SYSTEM ARCHITECTURE

Deep Space Station 13 (DSS-13) is an experimental beam waveguide antenna used by the DSN to investigate the applicability of Ka-Band deep space communications. The core subsystems in place at DSS-13 are a mixture of several generations of communications and radiometric instruments. The one characteristic they all shared at the beginning of the DSS-13 M&C development was that all subsystems were designed to work assuming the presence of a dedicated operator per subsystem. The success of the OSI testbed activity led to the concept of centralized M&C at DSS-13 using OSI standards and commercial-off-the-shelf (COTS) hardware and software. The goal set forth for the DSS-13 M&C team was to design, develop, integrate, test, and turn over to operations a centralized M&C system capable of supporting the Ka-Band Link Experiment (KABLE) using only one operator in slightly over one year.

The resulting M&C system is described in the following section. We also present a section on our approach to using the OSI standards, and the current status.

2.1 Station Monitor & Control System

The DSS-13 Station Monitor & Control System consists of a centralized monitor and control station (referred to as the Station M&C subsystem - SMC), the M&C interfaces present in each of the subsystems that are part of the DSS-13 core equipment, and the Local Area Network (LAN) which connects them all to each other. A block diagram of the DSS-13 configuration is given in Figure 1. The functional goals [Ref 2] of the M&C system are:

1. Provide a centralized M&C facility which will enable a single operator to operate all the equipment necessary to support KABLE.
2. Enable the SMC operator to perform any operations function for the subsystem through the SMC
3. Provide a means of distributing support data (for example, predict files) to the subsystems from the SMC
4. Provide a graphical user interface which can be tailored by station personnel to meet their operational needs.
5. Provide a means of recording and logging all operational functions, monitor data, and event notices.
6. Provide an open systems environment capable of supporting automation

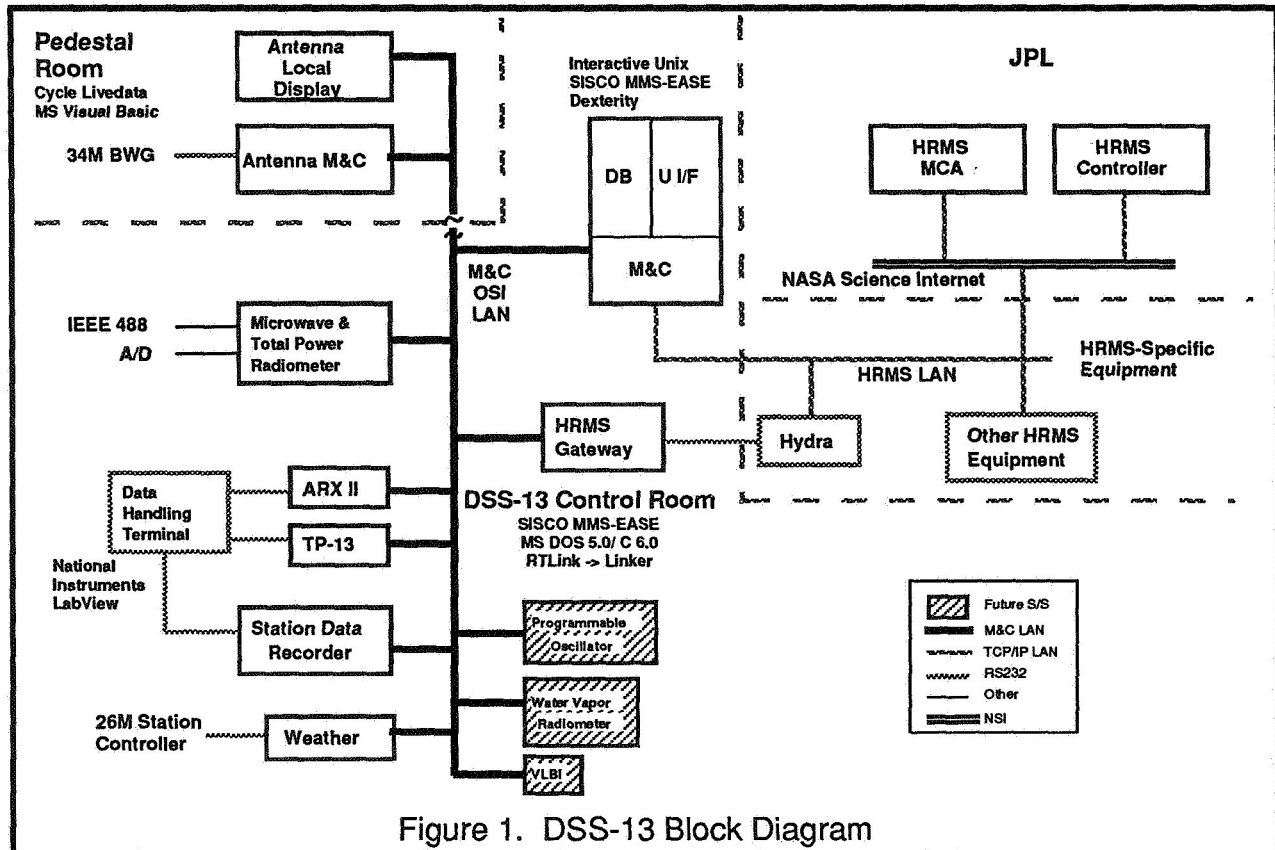


Figure 1. DSS-13 Block Diagram

Functional Requirement	MMS Services
Context Management	Initiate Conclude Cancel Abort
Client-Server Relationship	Initiate/Conclude Request Initiate/Conclude Indication Initiate/Conclude Response Initiate/Conclude Confirm
Distribution of Support Files	Copy Obtain Open Read Close Rename Delete
Directives, Displays, Events, Alarms	MMS Variable Access Read/Write Variable Information Report Get Attributes Define Type Delete Type etc.

Table 1. M&C Functional Requirements vs. MMS Services

The SMC system constraints were:

1. It must be flexible: enable new subsystems to be added quickly, easily, and inexpensively. Since DSS-13 is an experimental station, the ability to bring out new equipment, quickly connect to core equipment, and test out new concepts is an important part of its function.
2. It must be reliable: Since DSS-13 is also, in a limited sense, an operational station, the M&C system had to be able to routinely support operations functions
3. It must comply with the Government Open Systems Interconnection Profile (GOSIP)

2.2 Approach

The development of subsystems for DSS-13 was well underway when the concept of implementing a centralized M&C strategy at the station was adopted. Our approach to developing the SMC was highly influenced by the need to "retrofit" these subsystems with the capability to be operated over a computer network. In some cases, this was straightforward, in others, it required a significant effort on the part of both the M&C and the subsystem development

teams. In order to remain cost effective, our design decisions had to work with the existing hardware and software environments used by the subsystems with little or no modification. The following sections describe the basic design decisions made and their impact on the overall system effort.

2.2.1 MMS

The Manufacturing Message Specification (MMS) protocol was chosen as the standard on which to base DSS-13 M&C. MMS is an application layer protocol designed for process control. The OSI testbed activity demonstrated the solid match between the services offered by MMS and those required to support M&C operations functions, Table 1. Based on the commercial availability of products, the decision was made to implement the system using MMS over a Token Bus LAN. This decision enabled us to take advantage of the development work already done in the OSI Testbed and to begin immediate procurement of the necessary hardware¹.

2.2.2 Common Software and Environment

¹ Plans are already in place to replace the Token Bus with an ethernet.

Our approach centered on quickly standardizing a common language and operating system (in our case MS DOS and MS C, current versions). These were chosen because several of the subsystems were already using them, and because MMS products existed for these environments. The common software needed by each subsystem to form the interface between MMS and their applications was developed and integrated by the M&C team. In some cases, this required translating from one version of the C compiler to another, in other cases, it involved finding compatible products for other environments (e.g. Basic, UNIX). The integration effort was, at times, painful due to the "after-the-fact" decision to use a centralized M&C approach. The local control interfaces needed to be re-engineered, additional data structures were required to support additional feedback needs, and code had to be optimized to meet operating system memory size limitations.

2.2.3 Data Dictionary/Table

The primary interface between M&C and the individual subsystems is through the Data Dictionary. The Data Dictionary provides a listing of all the variables (used to send control information and monitor health, status, and performance) needed to run a subsystem. M&C contains a complete listing. Each of the subsystems contains a complete table of their own variables as well as additional data structures to provide specific information from other subsystems. The table-driven nature of the architecture enables rapid addition, deletion, and other changes and makes adding new subsystems relatively straightforward.

2.2.4 User Interface

The Dexterity² commercial user interface package was used to provide the SMC user interface. Draft displays were generated by the SMC team and given to the DSS-13 operators. They were trained in the use of the Dexterity package and began developing their own displays. In the DSS-13 environment where the types of activities being supported can change on a regular basis, it is very important for the operators to have the ability to tailor old displays and build new ones to meet their needs. Because of the open system architecture, data could easily be accessed through the data dictionary and displays incorporating a variety of parameters assembled without a software development effort being needed.

3. STATUS

The overall configuration for DSS-13 is as shown in Figure 1. The M&C system is currently being tested with all of the major subsystems required for KABLE operations. Development of the SMC began in October 1991, system integration with the DSS-13 subsystems in July 1992, and the station will be operational in order to support KABLE in January 1993. During the one-year development effort, two new subsystems, the Station Data Recorder and the High Resolution Microwave Survey (HRMS) gateway were added.

The HRMS gateway was developed to provide an interface which allowed the experiment to control station equipment over the M&C LAN when the previously planned direct connection was deemed unfeasible at the last minute. The subsystem was developed, integrated, and tested in under 4 months and supported the HRMS kickoff on October 12, 1992. The flexibility of the DSS-13 M&C architecture in responding to a short fuse customer request resulted in a successful experiment. In addition, the DSS-13 operators developed a display which allowed them to monitor the station equipment while it was under the control of the experiment and were able to detect a serious anomaly in time for the experiment to fix it during the actual experiment.

4. SUMMARY

The DSS-13 experience in the application of OSI standards to support M&C has been extremely successful. The chosen standard, MMS, meets the functional needs of the station and provides a level of flexibility and responsiveness previously unknown in that environment. The architecture is robust enough to meet current operational needs and flexible enough to provide a migration path for new subsystems.

The experience has also highlighted some important lessons learned on the application of COTS hardware and software and the importance of good systems engineering and configuration control. As a "breadboard" for a M&C system for the operational subnetworks of the DSN, the DSS-13 M&C System has provided valuable insights and a proof of feasibility for the overall architecture.

5. ACKNOWLEDGMENTS

The work described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. In addition to the authors, the DSS-13 M&C development team includes Mike Stockett, Richard Chen, Lyle Skjerve,

² Trademark, Nucleus Systems

and Bob Gosline. We would like to extend our appreciation to the numerous DSN operations and engineering personnel who have contributed to this effort, and especially Gerry Crook, Ron Littlefair, and Mark Gatti.

6. REFERENCES

1. Heuser, W. Randy, "An OSI Architecture for the Deep Space Network," Jet Propulsion Laboratory TDA Progress Report Volume 42-109, Pasadena, CA, January - March 1992.
2. Gatti, Mark S., and Lynne P. Cooper, "DSS-13 Conceptual Functional Requirements," JPL Internal Document IOM 3320-91-054, Pasadena, CA, February 6, 1991.

7. ACRONYMS

COTS	Commercial Off the Shelf
DSCC	Deep Space Communications Complex
DSN	Deep Space Network
DSS	Deep Space Station
FIPS	Federal Information Systems Processing Standards
GOSIP	Government Open Systems Interconnection Profile
HRMS	High Resolution Microwave Experiment
ISO	International Standards Organization
KABLE	Ka-Band Link Experiment
LAN	Local Area Network
MAP	Manufacturing Automation Protocol
M&C	Monitor & Control
MMS	Manufacturing Message Specification
OSI	Open Systems Interconnection
SMC	Station Monitor & Control
TDA	Telecommunications and Data Acquisition
VLBI	Very Long Baseline Interferometry

ATOS-1: DESIGNING THE INFRASTRUCTURE FOR AN ADVANCED SPACECRAFT OPERATIONS SYSTEM¹

K J Poulter and H N Smith

N 94-23849

Logica Space and Communications Limited and Logica Cambridge Limited
Stephenson House, 75 Hampstead Road
London, NW1 2NT
UK

1. An alternative title adapted from related work in the area of concurrent engineering (Ref. 1) is "Towards a Medium for Collaborative Space Mission Design and Operations"

ABSTRACT

The space industry has identified the need to use artificial intelligence and knowledge based system techniques as integrated, central, symbolic processing components of future mission design, support and operations systems. Various practical and commercial constraints require that off-the-shelf applications, and their knowledge bases, are reused where appropriate and that different mission contractors, potentially using different KBS technologies, can provide application and knowledge sub-modules of an overall integrated system. In order to achieve this integration, which we call knowledge sharing and distributed reasoning, there needs to be agreement on knowledge representations, knowledge interchange formats, knowledge level communications protocols and ontology. Research indicates that the latter is most important, providing the applications with a common conceptualisation of the domain, in our case spacecraft operations, mission design and planning. Agreement on ontology permits applications that employ different knowledge representations to interwork through mediators which we refer to as knowledge agents. This creates the illusion of a shared model without the constraints, both technical and commercial, that occur in centralised or uniform architectures. This paper explains how these matters are being addressed within the ATOS programme at ESOC, using techniques which draw upon ideas and standards emerging from the DARPA Knowledge Sharing Effort. In particular, we explain how the project is developing an electronic Ontology of Spacecraft Operations and how this can be used as an enabling component within space support systems that employ advanced software engineering. We indicate our hope and expectation that the core ontology developed in ATOS, will permit the full development of standards for such systems throughout the space industry.

Key Words: Knowledge Sharing, Ontology, Knowledge Based Systems, Mission Information Base, Spacecraft Operations, Mission Design and Planning

1. THE ADVANCED TECHNOLOGY OPERATIONS SYSTEM

During the past few years the European Space Operations Centre (ESOC) of the European Space Agency (ESA) have carried out a number of projects to demonstrate the feasibility of using advanced software technology, in particular artificial intelligence techniques and knowledge based systems, to support spacecraft operations. Although these applications have been successfully tested in isolation and for selected or simplified subsets of required mission functionality, there are a number of advances which must be achieved before AI techniques can be fully exploited in future mission systems, namely, *integration* of the applications into a single system with *consistency* in information model and user interface, together with *generalization* of the applications so that they are mission independent. Unless generalisation is achieved, the cost of development of such systems will be a stumbling block to operational deployment.

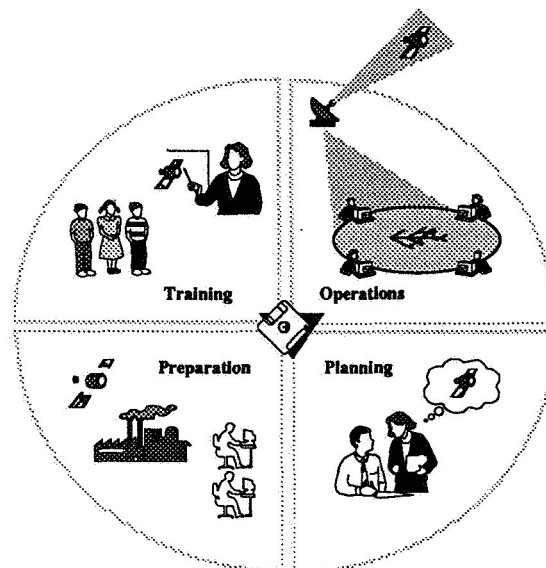


Figure 1 The Advanced Technology Operations System (ATOS)

In order to achieve this goal ESA has initiated the Advanced Technology Operations System programme (ATOS) which will develop the system concept and infrastructure to support spacecraft operations systems which utilize advanced software technology and the development of advanced application modules (AAMs) to support: Mission Preparation, Mission Planning, Computer Assisted Operations and Advanced Training.

Logica is leading the first project in the ATOS programme, ATOS-1, which is tasked with the goal of designing and prototyping software to demonstrate the overall system concept and to provide the formal specifications of the ATOS technology infrastructure, which is required as inputs to the other ATOS phases. Key to this endeavour is the formal specification of an *Ontology of Spacecraft Operations*. An ontology is a formal specification of the knowledge structures that can be used to model a domain, in our case the domain of spacecraft operations. Terminology and relationships between entities in the domain are explicitly defined. This paper explains the role of ontology in achieving ATOS goals. We believe that the approach described has wide applicability in acting as a catalyst for the exploitation and development of advanced mission support applications throughout the space industry and elsewhere. Recognising the significance of agreement on definitions is the key to moving knowledge based systems off the side lines and into mainstream mission support and operations. The aim is to create a medium in which collaborative mission design and operations can flourish.

As mission lifetimes extend and mission objectives and payloads become increasingly complex, advanced functionality must be made available to mission staff, to help manage the complexity and to provide efficient and accurate knowledge transfer between mission phases and between mission staff. ATOS technology is needed right now, but must be provided in a form acceptable to users. ATOS concentrates upon providing mission support power tools, rather than applications aimed at removing close human involvement in the mission operations process.

1.1 The Benefits of ATOS

ATOS aims to improve the operations viewpoint, the mission viewpoint, the human viewpoint and the technology viewpoint.

Operations require improved user interface facilities, particularly in terms of representation and manipulation of spacecraft models and operational procedures. Support to the user must include on-line documents, intelligent assistance and improved procedural reliability and thoroughness.

The mission requires efficient configuration, harmonisation of practices and minimisation of manual interfaces between currently disparate activities. The mission requires more flexibility in the allocation of manpower, tasks and skills.

The technology perspective dictates flexibility in the allocation of computer resources to enable distributed and scalable computing, together with a strategy that will provide open reusable and portable software applications.

All these matters will be addressed in the ATOS programme. The current phase, ATOS-1 is concentrating upon the knowledge that comprises a mission and is finding ways to store it, preserve it, reuse it and communicate it. This is known as *Knowledge Sharing*, which is a new technology for the 1990's which will enable a new range of sophisticated applications to be developed that exploit and interchange *knowledge as a valuable asset*.

The ATOS-1 project began work by looking at various initiatives in the formalisation of standards for platform level portability of knowledge. It soon became apparent that this in itself would not achieve ATOS goals and recent work has focused upon knowledge level (Ref. 2) representations and on the construction of ontologies, which formally define the knowledge structures that comprise a domain, in our case, spacecraft operations.

Further information regarding the motivation for ATOS, current activities in the programme and expected future work can be found in a companion paper (Ref. 3).

2. THE NEED FOR KNOWLEDGE SHARING, REUSE AND INTEGRATION

2.1 The Nature of the Problem

The space domain, like most others, has seen the development of a wide range of applications of knowledge-based systems during the past few years. Although many of these applications were only experimental prototypes, an increasing number are being deployed operationally (Ref. 4) or are entering pre-operational trials (Ref. 5).

In general, building a knowledge-based system involves developing a new knowledge base from scratch. This has been true of prototype applications in the space industry, with different groups building applications for the same agency, or contractors developing their knowledge bases in isolation. The duplication of effort involved has already been significant and will be a major stumbling block to operational deployment as application demands require larger and larger systems to be built. It is therefore vitally important that we establish a mechanism to enable knowledge to be reused within related applications. Operational versions of such applications will be considerably more complex than the prototypes built to date and include much larger bases of knowledge. This underlines the need for reuse of that knowledge and the applications which use it.

In addition to *reuse of knowledge* there are two further significant impediments to the development and de-

ployment of major KBS-based spacecraft operations systems:

- **Knowledge sharing;** allowing separate physical knowledge-based components within an overall system to share knowledge. For example, a diagnosis application and a monitoring application need to share the same overall model of system structure and behaviour, but may need to use different knowledge representations.
- **Integration of knowledge-based applications;** allowing the run-time exchange of knowledge between applications. For example, allowing an application to notify others of new things it has deduced as a result of changes in the operating environment.

It is finding solutions to these three factors which will open the door to a new generation of spacecraft operations systems incorporating advanced technologies.

Other aspects of knowledge based system technology, such as validation and verification are important but, in the context of the ATOS-1 project must be seen in a secondary role. Such activities are generally associated with KBS methodologies which are used to structure application development and guide the process of domain expert knowledge capture. These aspects will be important to future ATOS phases but are not a prime concern in ATOS-1, which must define the knowledge structures and semantics (ie not contents) to be used by applications to enable knowledge sharing and to decouple knowledge from inference mechanisms, the latter being a major source of impediment in achieving knowledge sharing in previous generation KBS systems.

2.2 Foundation for a Technological Solution

The problems identified with respect to reuse, sharing and communication of knowledge within the ATOS-1 project can be addressed by providing technological solutions in the following areas.

2.2.1 Knowledge Interchange

If a knowledge-based system is to make use of an existing knowledge base or library, or to interchange knowledge with another system, then the knowledge must either be encoded in the system's representation or be translatable into that language.

Given that for many applications practical efficiency is made possible by using a special purpose representation, it is impractical to expect a universal representation language to become available which will meet the needs of all space applications. Two different applications may wish to represent the same knowledge using different structures in order to improve the efficiency of reasoning in each.

The major initiative in this area is the Knowledge Interchange Format (KIF) (Ref. 6) which is being devel-

oped within the DARPA Knowledge Sharing Effort (Ref. 7). KIF has been developed for use as an interlingua (ie intermediate language) for communicating knowledge between computer programs which use different knowledge representations.

Within the ATOS-1 project we are adopting KIF as our knowledge interchange language.



Figure 2 KIF as an interlingua

2.2.2 Standardised Knowledge Representations

KIF is not intended to be used as a representation for direct manipulation by a computer system (although it can be). A further significant problem therefore to achieving practical knowledge sharing at present, even within close families of languages, is therefore that for each representation paradigm used by applications, eg. frames and semantic networks, there are a wide range of practical implementations of that paradigm. Each implementation has its own idiosyncrasies, which means that knowledge represented using one variant is difficult to share with the user of another.

This aspect of knowledge sharing and reuse is being addressed by the development of standards for knowledge representation languages, such as the IMKA initiative (Ref. 8) and the KRSS initiative within the DARPA Knowledge Sharing Effort (Ref. 7), which are working towards standards for frame-based representation systems and KL-ONE derivatives (Ref. 9) respectively.

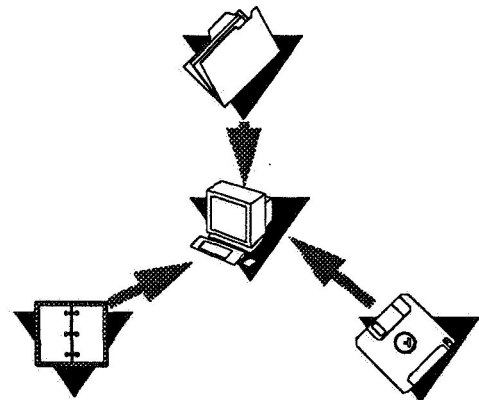


Figure 3 The need to standardise within a representation paradigm

2.2.3 Communication of Knowledge

A further aspect of the problems faced by ATOS with respect to knowledge sharing is enabling the run-time communication of knowledge between programs, ie. Enabling programs to exchange knowledge and to query each other with respect to the state of their knowledge bases.

Within the research community the major development in this area is KQML (Knowledge Query and Manipulation Language) (Ref. 10). KQML is a high-level language and a set of protocols which can be used by software systems for the run-time sharing of information and knowledge.

KQML is already being used as a knowledge communication in two testbeds: the Palo Alto Collaborative Testbed (Ref. 1) which is in the domain of concurrent engineering and the DARPA/Rome Planning Initiative which is concerned with military transportation planning. Within ATOS-1 we intend to develop a spacecraft operations testbed which will investigate the applicability of KQML to the domain of mission control.

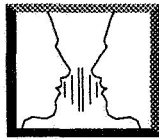


Figure 4 Communication at the knowledge level

2.2.4 Shared, Reusable Knowledge Bases

Despite the practical problems which can be overcome using developments such as KIF and KQML there still remains a fundamental problem with respect to knowledge sharing: the semantic *content* of a knowledge base. This requires that developers, and hence the applications they develop, have a common conceptualisation of the domain in which they are operating. That is, they must model those things which are important to their task in terms of the same objects, processes, relationships etc. Such a model is called an ontology.

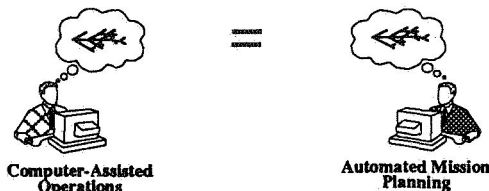


Figure 5 A common ontology in the minds of users and machines

In adopting a certain set of concepts and their relationships a program makes a number of ontological com-

mitments to a particular conceptualisation of the domain. If a number of applications adhere to the same conceptualisation they are said to have a *common ontology*. It must be stressed that in defining an ontology one is attempting to explicitly capture the formal semantics of a data model. This is therefore fundamentally different from the aims of the various standards being set in place for integration of object and database systems, which typically concern the format and usage conventions of information rather than the meaning of the knowledge implied by the information.

A major objective of the ATOS-1 project is to establish a common ontology for use by applications in the spacecraft operations domain, ie. *an Ontology of Spacecraft Operations*. This is discussed further in section 4.

3. AN OPERATIONAL CONCEPT

The ATOS objective is the establishment of an infrastructure by which intelligent knowledge based applications (KBS) interact at the *knowledge level*, that is they interact with respect to an explicit shared and agreed model of the domain of spacecraft operations. This idea is illustrated in figure 6:

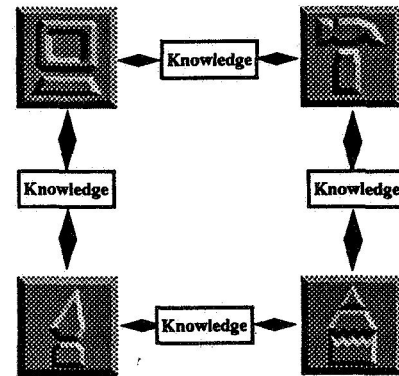


Figure 6 Applications interacting at the knowledge level

Since applications are unlikely to be developed according to a completely consistent common ontology and use identical representation systems, this can only be achieved if the applications interact through a *Distributed Access Service* (DAS) based on cooperating knowledge agents that themselves adhere to a consistent ontology of spacecraft operations. The complexity required within each agent will depend upon how far the applications deviate from the ontology. Agents will perform functions such as translation of query forms as expressed by applications into the standard ontological model. This idea is illustrated in figure 7:

Knowledge agents will mediate between disparate applications that employ heterogeneous knowledge representations, reasoning methods and query forms. The shared ontology that forms the fabric of the infrastructure will provide the means by which applications will access a logically unique repository of mission infor-

mation, the so-called *Mission Information Base* (MIB). The MIB itself is likely to be highly distributed and itself comprise heterogeneous representation systems (relational, object oriented, logic-based). In certain cases applications will themselves retain MIB partitions locally for specific purposes, with the express aim of optimising knowledge manipulation and ensuring that the most appropriate representation for the specific reasoning task is used.

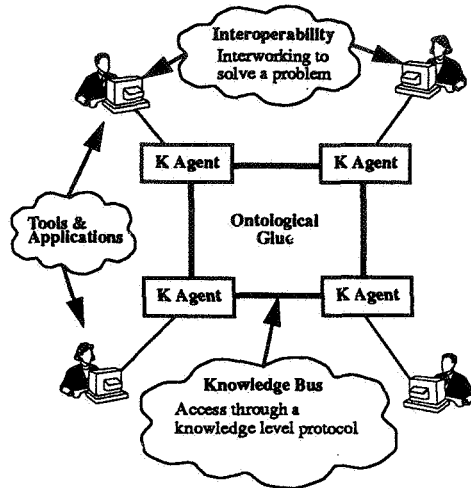


Figure 7 A distributed access service

Irrespective of the location of knowledge resources and of reasoning capability, each application will have access to the entirety of all mission knowledge and be able to call upon the deduction capabilities of all applications through the medium of knowledge communication. The shared ontology that comprises the fabric of the architecture will ensure that the semantics of mission knowledge are consistent between all applications.

The creation of such an infrastructure for a wider domain would be a formidable undertaking indeed. It is our view however that the domain of spacecraft operations, and current ESOC required functionalities, is of sufficiently contained scope to ensure that the development of the core ontology remains a tractable problem.

ATOS applications that adhere to the ontology will be *generic, mission independent, human engineered tools*, tailored for specific mission design or operational tasks. Examples include planning and scheduling, diagnosis and trouble shooting, monitoring and control, engineering modelling and simulation, schedule and procedure execution.

4. THE ONTOLOGY OF SPACECRAFT OPERATIONS

The ATOS-1 project will establish a core ontology of spacecraft operations. This will define the objects, re-

lationships, processes etc. required to represent core knowledge in the spacecraft operations domain.

The core areas which are being covered within the ATOS ontology include: Planning and Scheduling, Engineering Modelling, Monitoring and Control, Diagnosis and Troubleshooting, Flight Dynamics and User Interface Model. Within each area a number of sub-topics have been identified and these will form logically self-contained units of ontology, called theories, which can be used by applications in a discrete fashion if required.

```
(define-class system (?s)
  "System is the root class of all systems. A system may be a
  primitive object or consist of a number of other systems. It doesn't
  make sense to allow instances of system itself so it is constrained to
  be an individual."
  :def (individual ?s)) ;from KIF ontology - not a set

(define-relation part-of (?s1 ?s2)
  "s1 is a part of s2. The part-of relation is limited to a system
  and its immediate subsystems, ie. It is not transitive. This helps to
  view the overall system at differing levels of abstraction which can
  be useful in modelling to limit the detail to that required in a partic-
  ular situation. A system cannot be a part of itself, ie. Irreflexive."
  :def (and (system ?s1)
            (system ?s2))
  :axiom-def (and (not (transitive part-of))
                  (irreflexive part-of)))

(define-relation components (?s ?c)
  "Components relates a systems to its direct components, ie.
  The systems ?S for which (part-of ?S ?s) holds. For a system which
  is a primitive object components is the empty set. A system cannot
  be a component of itself, ie. Irreflexive and a system cannot be a
  component of a system that is a component of it."
  :def (and (system ?s)
            (system ?c))
            (part-of ?c ?s))
  :axiom-def (and (not (transitive components))
                  (irreflexive components)))
```

Figure 8 Example of ontology written in Ontolingua

In selecting a representation and tool for developing the ontology a number of requirements were addressed. The most significant requirements in the decision process were:

- **Well-defined semantics** - the ontology will form a resource to be used by many application developers, this means its interpretation must be unambiguous;
- **Electronically readable** - the ontology will need to be exchanged by a variety of developers

and worked upon independently at different sites;

- **Translatability** - the ontology will be required in many different forms, dependent upon the particular tool being used by the developer.

The representation that has been adopted is Ontolingua (Ref. 11). Ontolingua is a language and a supporting translation architecture which has been specifically designed for the development of portable ontologies. Ontolingua is being developed as part of the DARPA Knowledge Sharing Effort, unifying the predicate calculus representation of KIF (Ref. 6) with the concepts found in object-centred knowledge representation systems. KIF is based on predicate logic, with a formalised mathematical foundation. Ontolingua provides the means to structure assertions in object-centred hierarchies with inheritance via standard primitives for defining classes and relations.

An example of part of the spacecraft operations ontology is given in figure 8. It illustrates a class and two relations used to form ontological commitments between applications wishing to adopt a common model for structuring the spacecraft model hierarchically.

To help convey the idea of the ontology further it is worthwhile considering an alternative perspective from the application-oriented (ie by Ontolingua "theory") partitions discussed earlier. This is to view the ontology as consisting of layers of increasing specialisation of generic abstract concepts such as "process", "event" etc. These are specialised firstly so as to form abstract domain concepts, such as system, device, sensor etc, and then specialised into generic spacecraft operations concepts, eg. telecommand, solar panel etc. Finally this is specialised in terms of any agency-specific terminology or conventions. This is illustrated in figure 9.

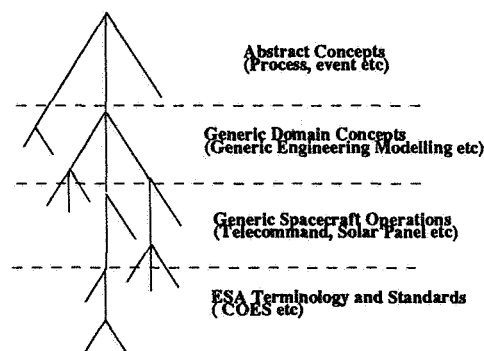


Figure 9 Layers of increasing specialisation in the spacecraft operations ontology

5. A MEDIUM FOR COLLABORATIVE MISSION DESIGN AND OPERATIONS

When the ATOS programme was conceived, four applications were identified as comprising the most promising areas for exploitation of AI technology in a space mission support system. These are: Mission Preparation, Mission Planning, Computer Assisted Operations and Advanced Training. They remain concrete target areas for ATOS application. It is unlikely however that these sub-domains will be realised as discrete components within the ATOS infrastructure. "Cutting" the ontology in this way may well prove to be inefficient. A more natural ontological division, which minimises the need for interaction between applications, may be more appropriate. The ontological analysis being conducted in ATOS-1 may suggest a different partitioning of the domain and may conclude that a larger number of smaller applications would be more appropriate. Unless this matter is addressed, the undisciplined interchange of knowledge between applications could be too unwieldy. One must attempt to minimise the ontological commitments between applications in order to maximise the independence of these applications. This results in minimising the need for cooperation between them. In this way the architecture will be open to the greatest possible extent. Shared ontology should not be viewed as a constraining factor, but as a mechanism to permit the *integration of disparate views of the domain and of methods for problem solving*. For example, an application which is modelling a device mechanically views the device differently from an application modelling the electronics controlling the mechanical linkages. Both applications however, if they are to cooperate, need to share some aspects of the model. This is where ontology is crucial.

In making the comments above it is important to distinguish between knowledge sharing, which is the placement of knowledge within tools and information bases for direct access by those tools, as contrasted against knowledge communication (as typified by KQML), which is concerned with the dynamic aspects and inter-relationships between knowledge "islands". It is the latter where it is desirable to minimise ontological commitments.

PACT (Ref. 1) points out that "in current infrastructures, applications do not really interact with respect to a shared model: the users do, but the applications do not know this is happening". A group of users working on a mission each have their own tasks, goals and domains, but they nevertheless have a shared view of the evolving mission and spacecraft design. ATOS technology attempts to capture this as knowledge. The evolving mission design, the representation of the spacecraft, its operating procedures and mission objectives, are represented only very superficially in current computer systems, if at all. Ontologies are not represented in any way within current implementations. The basic premise of ATOS is that we can move to a system in which applications themselves interact through a *shared explicit model of the mission*. This requires the establishment of a *knowledge level bus* into

which applications are plugged.

Spacecraft design, mission planning and analysis are “processes of negotiation; decisions are made and changed frequently as specifications change and new implementation ideas proposed. Applications must interact frequently, interchanging relatively complex information. The applications must assist the users in understanding and keeping track of each others constraints, dependencies, limitations and assumptions. Different users may be concerned, at any one instant, with different problems or different aspects of the same problem, or will be working at different levels of detail. Despite these differences in perspective they all share considerable information about the overall mission”. Enabling users to share this information and to coordinate the decision making is one of the problems being addressed in ATOS for the domain of spacecraft operations. The role of common ontology is crucial in enabling collaborative space mission design and operations. It is necessary for the applications to agree on many matters, including definitions of space, time, co-ordinate frames, units of measure, system structure and constraints, in order to effect translation between the different representations used by the applications.

Research has shown that application integration (ie agreement on the form of messages, as opposed to the content of messages), even across heterogeneous platforms, has turned out to be a relatively easy task. However, to enable knowledge sharing one has to tackle the far harder and more fundamental problem of agreeing on ontological commitments. This is due to the differences in abstractions and views employed by the different applications in areas such as planning, diagnosis, design and modelling. The shared language to which applications must adhere, couples these abstractions and *creates the illusion of a shared MIB model without the constraints and bottlenecks that occur in enforced centralised approaches*. This more loosely coupled architecture permits currently disjoint user teams to cooperate without the imposition of centralised technologies on existing work patterns, and the inevitable disruption of those work patterns. We view ATOS technology as more fundamentally practical, open and of sounder foundation than any centralised approach could achieve.

6. ATOS ARCHITECTURES

We intend to directly utilise the ontology developed in ATOS-1 in the construction of the architecture, firstly in the creation of the mission information base schema and also in coupling interfaces between ATOS applications. This idea is illustrated in figure 10:

There are three perspectives on how the ontology would be used to construct an ATOS architecture:

1. Translation into the representation schema employed within selected MIB and advanced application technologies.
2. To support the reuse of generic operations

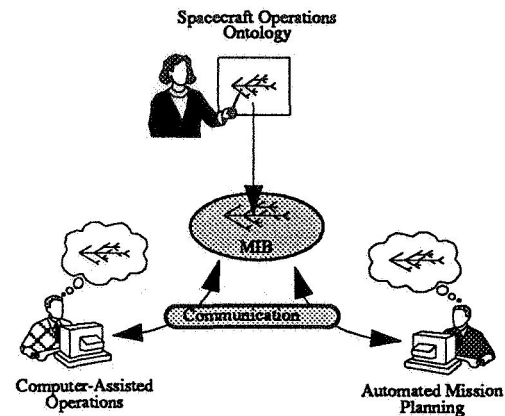


Figure 10 The role of the ontology in the ATOS architecture

knowledge across missions and across application implementation platforms in a form separate from its actual encoding within the MIB. We call this a knowledge resources base (KRB).

3. To develop a set of run-time communications protocols in order to enable different inference engines to reason using the knowledge stored in the MIB, a so-called knowledge query and manipulation language.

6.1 Architectural Perspective 1

Space permits only the first perspective to be described in this paper, which is illustrated in figure 11. The principal elements to consider with respect to this architecture are as follows:

- The common Spacecraft Operations Ontology is maintained as a collection of machine readable definitions in Ontolingua. Ontolingua is itself defined in terms of KIF.
- The common ontology is effectively a reference source of knowledge structures which are deployed into parts of the ATOS system itself. The Ontolingua definitions themselves do not form part of the final system. The ontology is translated (semi) automatically into other parts of the system, so as to establish a common model of the domain throughout ATOS.
- The MIB contents definition is derived by translating the ontology into the MIB information or knowledge model. This is an off-line process and happens only once during the development of an ATOS-based MCS. The MIB's information or knowledge abstraction may be provided directly by a specific commercial tool or layered upon one. In the later case the MIB platform will be selected upon the basis of the flexibility of the storage management it provides rather than the

expressive capability of its data model.

- Advanced application modules (AAMs) will need to share the common ontology in order to “understand” the relevant contents of the MIB. If an AAM is to reason using knowledge stored in the MIB this is also a prerequisite. For example, a planning tool, will need to know whether the MIB calls the results of successfully executing an activity a “postcondition”, or an “effect”.

The ontology (or relevant parts of it) will be translated into the KBS tool which is being used for developing the AAM. For example, this would consist of a collection of classes and their associated slots which are equivalent to the Ontolingua classes and relations defined in the ontology.

The translated ontology provides a set of definitions for the application developer to build his implementation in terms of, for example, in the planner example given above definitions of activity, procedure, temporal constraint would be provided. Application knowledge will be created by instantiating the classes etc derived from the ontology or by extending the definitions with any further knowledge structures which are required specifically to support the task of the AAM.

As with the MIB, the translation process is off-line and occurs only once during the development of an AAM.

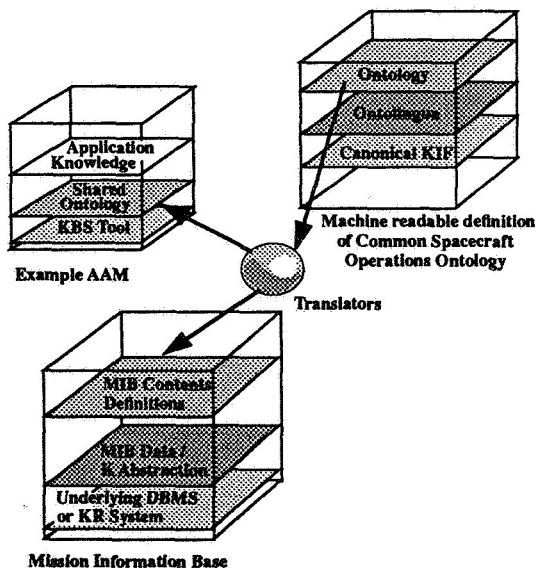


Figure 11 Architectural perspective 1

7. TECHNOLOGICAL FOUNDATIONS OF ATOS

Knowledge sharing, reuse and communication are problems in a wide variety of application domains, not just spacecraft operations. In setting out to address these technological problems within the ATOS programme we have made a conscious effort to follow researchers in other domains and to benefit from fundamental research which is addressing the underlying technical issues.

A particular influence on our technological approach is the work being carried out within the DARPA Knowledge Sharing Effort (Ref. 7). This initiative is working towards the establishment of standards for knowledge interchange (KIF), interoperability between intelligent agents (KQML) and tools to support the development of common ontologies for particular domains (Ontolingua).

With respect to ontology, there are a number of sources of information which are guiding and influencing its development. These are:

- Previous research in knowledge-based applications, in particular those in the areas of planning and scheduling, model-based reasoning and qualitative modelling. In particular, work which has a sound theoretical foundation. Although much of this previous work does not formally define the ontology it is adopting, reusable ontological fragments can be extracted with careful analysis. For examples of such sources, see Ref. 12, 13, 14 and 15.
- On-going research within the DARPA Knowledge Sharing Effort on the development of common ontologies for a number of domains.
- Previous applications in the space domain which serve to identify the domain specific knowledge structures and specialisations which are needed to use the more generic aspects of the ontology in the ATOS applications.
- Work at ESOC in the establishment of standards for ground based spacecraft control systems, and in particular the work of the Committee for Operations and EGSE Standards (COES). For examples see Ref. 16 and 17.

8. POSSIBLE FUTURE DIRECTIONS FOR ATOS

We hope that the core ontology of spacecraft operations, being developed in ATOS-1, will be able to evolve into a concrete standard which will act as a lever to enable the development of a range of advanced mission support and operations applications deployed within an integrated mission control system. The ontology, being expressed using a formal representation explicitly designed for the specification and export of

portable ontologies, permits its direct use in the construction of such applications and their knowledge bases. The mechanism for this will be the automatic, or semi-automatic, translation of the ontology into other representation systems employed by these applications. In support of this, the ATOS programme could become the coordinating initiative for the distribution of the ontology and could make available skeletal translation environments to other representation systems. We see the need for the following:

- To establish the means by which the core ontology being developed in ATOS-1 can be further pursued outside of the project, including funded experiments aimed at verifying ontological commitments between planned application domains.
- To establish the means by which the ontology can be transferred, with its translation environment, to parties authorised to utilise the knowledge resources being developed in ATOS-1.
- To establish an advanced interfaces initiative aimed at specifying the standard knowledge bus into which third party applications can be plugged, together with the means to verify the correctness of function of those applications with respect to the ontology. This is similar to that needed for communications protocol verification.
- To establish practical projects, aimed at developing wanted applications (not prototypes), which adhere to the ontology and which concentrate upon identification of minimum ontological commitment between sub-domains.

In particular, we recommend that the continuation of the ATOS programme, in parallel to and beyond ATOS-1, could follow a collaborative model similar to the PACT initiative. We believe it would be quite practical to establish core knowledge bus facilities, operating over a wide area network (eg the InterNet), in order to couple development and research teams working at different sites on different aspects of ATOS technology and its application. The aim would be to bring together other contractors, who have already contributed to furthering the use of AI in space operations in pre-ATOS prototype studies at ESOC, in a way which permits technology transfer and the further coordinated development of the ontology and mission information base.

Recognising the significance of agreement on definitions is the key to moving knowledge based systems off the side lines and into mainstream mission support and operations. The aim is to create a medium in which collaborative mission design and operations can flourish.

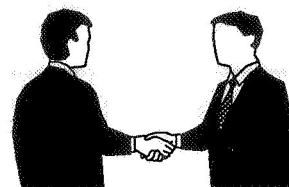


Figure 12 The answer

9. CONTACT DETAILS

The first issue of the Ontology of Spacecraft Operations, which must be considered a "draft proposal", is to be available in July of 1993. Requests for copies of this document should be made to Mr. H A Laue at the European Space Operations Centre, Darmstadt, Germany. We invite comment on the ontology from authorities and contractors across the space industry.

Eventually we also see the need, and will support, the establishment of a forum for common discussion regarding the ontology, the members of which will collaborate on its further development and exploitation. Informal communication can be established now by contacting the authors of this paper at the following email addresses:

howards@logica.co.uk

kevinp@logica.co.uk

10. ACKNOWLEDGEMENTS

The authors would like to acknowledge the significant influence that the results of the DARPA Knowledge Sharing Effort have had in the direction of this work. In particular, Tom Gruber, whose personal communications have helped to guide us through the early stages of ontology development.

11. REFERENCES

1. M Cutkosky, R Englemore, R Fikes, T Gruber, M Genesereth, W Mark, J Tenebaum, J Weber, "PACT: An Experiment in Integrating Concurrent Engineering Systems", Submitted to IEEE Computer January 1993 (Special Issue on Computer Supported Concurrent Engineering).
2. Newell, A, "The Knowledge Level", Artificial Intelligence 18, 1982.
3. Laue, H, Kaufeler, J-K, Poulter, K, Smith H, "The Advanced Technology Operations System ATOS", Proc. 2nd International Symp. 'Ground Data Systems for Space Mission Operations', SPACEOPS 1992, 17-20 November 1992, Pasadena.
4. Zweben, M., "Space Shuttle Processing: A Case Study in Artificial Intelligence", 28th Space Congress, 1991.

5. M Nielson, K Grue, B Olalainty, F Lecouat, J Wheadon, Expert Operator's Associate (EOA), An Expert System for Spacecraft Control, Proc. ESA Symp.: 'Ground data systems for spacecraft control', Darmstadt, FRG, 26-29 June 1990 (ESA SP-308, October 1990).
6. Genesereth, M. et al "Knowledge Interchange Format Version 3.0 Reference Manual", Report Logic-92-1 Computer Science Department, Stanford University, June 1992.
7. Neches, R et al, "Enabling Technology for Knowledge Sharing, AI Magazine, Fall 1991.
8. IMKA, "Phase 1 Software Functional Specifications", July 1990.
9. Brachman, R.J. and Schmolze, J.G. "An Overview of the KL-ONE Knowledge Representation System, Cognitive Science 9, 1985.
10. KQML Advisory Group, "An Overview of KQML: A Knowledge Query and Manipulation Language", March 1992.
11. Gruber, T., "Ontolingua: A Mechanism to Support Portable Ontologies, Version 3.0", Technical Report, Knowledge Systems Laboratory, Stanford University, June 1992.
12. Allen, J. and Hayes, P., "A Common-Sense Theory of Time", Proc. IJCAI-85.
13. Falkenhainer, B., and Forbus, K.D., "Compositional Modelling: finding the right model for the job", Artificial Intelligence 51, 1991.
14. Forbus, K.D., "Qualitative Process Theory", MIT Artificial Intelligence Laboratory, Technical Report 789, 1984.
15. Miller, D. P., "Planning by Search Through Simulations", Yale/CSD/RR/#423, October 1985.
16. ESA/ESOC, Committee for Operations and EGSE Standards, EGSE and Mission Control System (MCS) Functional Requirements Specification, PSS-07-401 Issue 1, Draft 6, April 1992
17. ESA/ESOC, Committee for Operations and EGSE Standards, Spacecraft Operations Interface Requirements Document (SOIRD), Issue 3, October 1990.

Distributed Science Operations for JPL Planetary Missions

Richard D. Benson and Peter B. Kahn

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

N94-23850

ABSTRACT

Advances in spacecraft, flight instruments, and ground systems provide an impetus and an opportunity for scientific investigation teams to take direct control of their instruments' operations and data collection while at the same time, provide a cost effective and flexible approach in support of increasingly complex science missions. Operations of science instruments have generally been integrated into planetary flight and ground systems at a very detailed level. That approach has been successful, but the cost of incorporating instrument expertise into the central mission operations system has been high.

This paper discusses an approach to simplify planetary science operations by distributing instrument computing and data management tasks from the central mission operations system to each Investigator's home center of observational expertise. Some early results of this operations concept will be presented based on the Mars Observer (MO) Project experience and Cassini Project plans.

1. INTRODUCTION

The Jet Propulsion Laboratory (JPL) has been conducting the mission operations of planetary exploration spacecraft for 30 years. The typical flow of work reflects processes that have been proven effective and reliable, various solutions to historic problems, and a mature organizational structure. Generally, spacecraft expertise has been developed at JPL while the science instruments originate at various investigator sites. Conduct of mission operations has often been supported at the most detailed level by the central JPL operations organization to maximize the density and quality of science data return and to ensure error-free operation of the spacecraft system.

The goal of a distributed mission information system architecture is to significantly improve the cost effectiveness and flexibility of mission operations. This is essentially a re-engineering task which unleashes and distributes computing power to enable processes to be accomplished more simply. The clearest cost saving areas are: 1) the cost avoidance of relocating science instrument and/or investigation teams to JPL for the mission duration (invariably years long and very disruptive of careers and families as well as expensive) and 2) cost avoidance of translating instrument and investigation knowledge from developer and user organizations to a separate, centralized operations organization. In a distributed system, the scientists are able to conduct most project-related science instrument operations from their home institutions via networked connections to a mission information system.

Improved flexibility comes from partitioning tasks and resources along investigation lines so that changes in investigation objectives and observing tactics can be done with comparatively minimal coordination with the central system. One benefit to a distributed operations system is that it provides for faster Science team response to changing instrument conditions. The data are provided to those with the most expertise and knowledge of the instrument without incurring significant time delays that would otherwise be imposed in a more centralized system.

MO is first JPL mission to use a distributed approach for science operations. This approach provides a unique "telescience" capability for the science instrument teams to remotely generate the observational uplink sequences and to provide near real-time analysis of the health and welfare of their instruments. Within the Space

Science community this is not a unique capability. The recent STS-52 flight of the Space Shuttle Columbia made use of telescience capabilities. Other missions, Hubble and UARS for example, have also utilized distributed science operations.

Cassini can utilize the experiences gained by MO and apply them to their unique needs to improve overall system effectiveness and flexibility. The Cassini mission will expand on this distributed approach capability for its operational requirements. Both missions involve significant on orbit time as opposed to one time flyby missions, so both can take advantage of the fact that observing opportunities are not once in a lifetime events but can be repeated on subsequent orbits.

Many operations tasks, especially system-level tasks, are more efficiently done centrally. However some subsystem functions are best distributed due to the 1) decreasing cost of computational power, putting more capability in the hands of the user, 2) the widespread use of communications infrastructures such as local and wide area networks and 3) financial constraints to keep staffing levels down and making better use of "smart" systems.

2. END TO END DESIGN CONCEPTS

Information system engineers at JPL try to think in terms of data processing in a true system sense. The central idea in "end to end" system engineering is to view processes and interfaces from an overall perspective to guide instrument, spacecraft, Deep Space Network (DSN), and Mission Control Center engineers toward designs that increase effectiveness and reduce costs by taking the end-user needs into account. Often this has the simple, but significant payoffs of eliminating redundant processes and promoting interface compatibility by making internal processing choices in favor of compatibility over expedience. An end to end view can also ease the development to operations transition of a project. Often that leverage is harder to apply because the phases are usually funded by different sources.

We see distributed science operations as a natural continuation of the end to end

approach. We discuss six areas where distributed science operations advance or are supported by the end to end approach. First is the delegation of function to whatever node is the best position to apply expertise directly. Second is processing, often applications software, that enable system and subsystem users to contribute to information processing tasks. Finally are database, standards, archiving, and security processes that supply an enabling infrastructure.

2.1 Decentralization of Science Operations

Cassini distributed science operations seeks to build on the experience of MO, which is the first JPL mission to use distributed instrument operations along with standard data interchange formats and multi-mission services to reduce project costs. There are differences in constraints between these two Projects, especially on the uplink processing. The distribution of tasks and trade-off analysis is customized for each mission. However, there are significant similarities in processing that lead to adoption of a distributed data system. Both missions take a hierarchical approach to operations. Both make significant use of multi-mission resources to build their uplink and downlink processes. Both distribute their operational processes and support to numerous teams, including remote science teams. MO has seven principal and six interdisciplinary investigations. Cassini has 12 principal and six interdisciplinary investigations. Most of these teams are not at JPL. The science teams further distribute the operational functions of the particular instrument to a variety of components within their own internal organization.

Some of the principles driving distributed operations are controversial. For example, the trend in many space science disciplines is to place observing decisions, including their technical and instrumental aspects, in the hands of observers who have operational expertise; this moves tasks away from historic, mostly centralized, venues.

Client-server architecture and widely available processing power have revolutionized data processing roles and made delegation of operational tasks to the

observer feasible and warranted. However, greater instrument/subsystem autonomy is not without some risk.

Packet command and telemetry standards allow for inheritance from mission to mission thereby decreasing development costs and time, but may involve more overhead (in terms of data volume) than custom formats.

The overall effect of the network and standards trends is to enable assignment of system functions to sites where they make the most sense. Certain functions that were done in centralized processes can now be delegated to the system's end users if it can be accomplished in a better, faster, cheaper process. Whether each function is done centrally -where common processes and system-level expertise can be applied, or distributed -where subsystem expertise can be directly applied, needs to be weighed. The emphasis of the distributed approach is on setting up a process that creates a satisfactory product in a minimum number of steps as opposed to a process that reshapes a product via rework into one that is ultimately satisfactory. Inheritance of design features from project to project decreases development and operational costs.

Coordinated efforts between the project mission operations teams, science teams, and multimission teams is essential for the design and operation of information data flow necessary to conducting planetary missions. MO had good success in hosting appropriate working groups. Cassini is just beginning to build such a working relation.

2.2 Processes

2.2.1 Sequence Design and Planning

The Science Investigation Teams are responsible for overall science operation (both scientific judgment and instrument operations aspects) because the requisite scientific, observational, and instrument expertise primarily reside there. Observing resources are allocated through a responsive, but systematic, process that resolves science conflicts at the science level based on science value judgments. In the end, all users are obligated to stay within their final allocations. The central

Ground System assigns reasonable margins to facilitate mission operations. (The term Ground System (GS) is used in the Cassini Project, and includes all applicable institutional and project-unique elements whose operation is centered at JPL. The GS includes the central functions in contrast to the distributed science functions.)

2.2.2 Command Generation

Instrument Commands - Detailed observation designs, both functional and performance aspects, are done by the Investigation Teams. Instrument-internal sequences/commands are generated, constraint-checked, and validated by the Investigation Team. The Ground System accepts these inputs as effective and reliable. The Ground System will not recheck these at a micro-level, in fact it generally can not. The GS may check them for compatibility with the system-level resources allocations and conventions depending on project-specific needs.

System Commands - The Ground System is responsible for mission success in an overall sense. This is a much broader role and includes providing: 1) a physical environment that enables the instruments to successfully operate (e.g., trajectory, pointing, thermal, radiation, and supporting services); 2) central sequence planning, integration, uplink processing, and orchestration; includes tools for conflict and constraint checking; 3) central data collection on board, accounting, and transmission to the ground; 4) central data reduction, correlation, and distribution on the ground.

2.2.3 Instrument Health Assessment

Overall system-level analysis is the responsibility of the Ground System. This includes analysis of the spacecraft subsystem performance, the space-ground link, and the GS itself. This analysis can include instrument assessments up to their interactions with the overall system. It should be noted that the GS itself is a distributed system with various project-specific and multi-mission components.

Instrument subsystem-level performance analysis is the responsibility of the science

teams, although many capabilities of GS tools and teams can be applied. The cost and utility of the GS tools needs to be compared to the cost of using investigation-specific tools, such as the instrument support equipment retained and upgraded from the instrument development phase.

By providing the capability for the remote teams to monitor their instrument parameters in near real-time, engineering analysis and response times are faster.

2.2.4 Science Analysis

Science data analysis to interpret the observations, to decide future observing tactics, and for final science data record production are the responsibilities of the Investigator.

2.3 Centralized Project Databases

A core component of a distributed data system is a mechanism that allows for transfers of data files between various Project elements. One such component is a Project Database (PDB). The use of commercially available database systems tailored to the specific needs for a Flight Project is essential for the dissemination of data between operational components of the Project. Use of such a system allows for managed distribution and archival capabilities. A PDB is built and maintained as part of the multi-mission capability of JPL. The inheritance of the concept is improved from mission to mission. The use of specifically labeled "Data Objects" enables science and engineering teams to store and retrieve data products in a centralized PDB and eventually archive them in the Planetary Data System. Both raw telemetry and file types of data are stored.

The PDB design is divisible into two primary components; a relational catalog and a database for telemetry and file data. The database catalog is the primary mechanism for the Flight Project science and engineering teams to determine what data are available in the database. All data sets must be sufficiently self-described in the catalog to allow the PDB to uniquely identify the product. The Science and GS teams can query the Database for values

associated with the predefined attributes of a data product, as listed in the Catalog. Interactive or scripted selection of a displayed list of associated attributes for any chosen data object can be accomplished. The user can select and query for data in a systematic way allowing for routine operating procedures.

2.3.1 Science Team databases

The Science Investigation team have, in many cases, created central databases or distribution systems at their facilities in order to supply recently-acquired instrument data to their team members and Co-Investigators at remote locations from the instrument operations teams. Electronic access to these facilities is provided, through a contract with the Project, by the NASA Science Internet (NSI). This is the primary distribution mechanism for intra-team science products during the life of the mission. Alternately, the data residing in these databases are copied onto some form a media and mailed to the team members.

2.3.2 Science community/National databases

A central catalog of archived planetary data is available to the Planetary Science community through a distributed system of discipline database archives. Flight Project archival data products generated using a common metadata labeling standard are available to the general Planetary Science Community through these discipline nodes. It is important that the catalog information used to query for the individual products be provided to these central databases along with the archival products themselves. Scientists from around the world can request data from all the existing and past Planetary missions for research. The requester queries the central database to find where the specified data are located. The users then submits a request for that data with the discipline node. Access to this system enables scientists not associated with the specific Flight project to participate in the evaluation and study of the data collected by the mission.

2.4 NETWORK STANDARDS AND COMMERCIAL TECHNOLOGIES

The distributed nature of a ground data system for science operations is made possible through the use of standardized, commercial technologies, such as, local and wide area computer networks, the Internet infrastructure, off-the-shelf network communication working protocols and UNIX-based computer operating systems. Instead of relying on large centralized Project support teams, the Science Investigation Teams, located throughout the country, can utilize these technologies to perform their own analysis of the data without significant time delays.

Compact Disk- Read Only Memory (CD-ROM) as a data distribution media is another area of commercial technology being adopted by JPL Flight Projects such as Viking, Voyager, Magellan and Mars Observer. Use of the international CD-ROM ISO 9660 standards along with data format and self-description standards continues the evolution of common Planetary Science interchange methods for heterogeneous data sets.

2.5 ARCHIVING

Early definition and design of a Project's archive process is essential to ensuring the smooth flow of data to the general science community. Once again, the distribution of this task from a central organization to the instrument operational sites increases the knowledge base applied to the data and expedites the product deliveries to the general science community. With the advent of CD-ROM technologies and software, each science operations team can produce and deliver archival data products directly to their discipline archive node without the need for large, centralized Project archive and distribution teams. Cost efficiencies are gained as is the flexibility of the individual science team to tailor it's products specifically for the end user community. Use of common mapping formats and time tags is essential for cross-correlation and interdisciplinary studies. Early planning of the archive process in the design of the Project insures commonality of format and structure.

2.6 SECURITY ISSUES IN TODAY'S ENVIRONMENT

One of the major issues from a cost and schedule perspective when designing a distributed operations system is system security. The emphasis and institutional sensitivity placed on computer network violations has increased significantly over the years. A Project needs to balance the need for open interchanges of information between distributed nodes and users vs. the fear of intrusion that can force limited access to mission data even for validated users. This risk assessment or trade-off analysis and the associated cost impacts of open vs. closed systems is a challenge in the era of distributed operations.

The area of most sensitivity to Flight Projects is the security of the uplink system. This system in which an Instrument team sends sequence files or commands to an instrument in flight, poses the greatest risk to mission objectives. Obviously, an unauthorized intrusion into this system places at risk the whole mission. Substantial checks and balances by both the remote instruments teams and the institutional GS teams is required to ensure the validity of the sequencing system.

An interesting dichotomy is that the Project receives services from the NASA Science Internet (NSI) to connect all their remote science users. This access to the national infrastructure, called the Internet, both increases accessibility to the data along with promoting Interdisciplinary studies. At the same time this access puts the Project at some level of risk of network intrusions by non-authorized users.

A risk analysis needs to be performed that balances the needs of the free interchange of ideas and information against the need to prevent unauthorized intrusion and possible destruction of mission critical data. Quick expensive hardware or software solutions may not provide security and can inhibit necessary science and mission interactions.

Table 1. DIFFERENCES BETWEEN MARS OBSERVER AND CASSINI

<u>Area</u>	<u>Mars Observer</u>	<u>Cassini</u>
Mission Plan	Circular, sun sync, repetitive orbits	Orbital tour, 60 orbits, each somewhat unique
Specific Targeting	No, S/C Boresight always points at nadir	Yes, boresight is under constant, specific control
S/C Operating Modes	Simultaneous observations and earth communications	S/C either points remote sensors at target or antenna at earth
Orbit activity level	fairly constant	High at encounter, lower at apoapsis
S/C Power	Very robust, large margins	Logical restrictions (Operational Modes) needed to avoid overdraw
Cruise activity	Instrument checkout, maintenance, calibration	Limited maintenance during cruise, No instrument calibration until Jupiter (June 2002)
Instrument Command Verification	Validated commands are verified by GS, but not functionally simulated	All commands may be re-verified by GS via system level simulation or analysis (to be determined)

3. COMPARISON OF MO AND CASSINI

MO is currently operating enroute to Mars while Cassini is in development for an October 1997 launch and June 2004 arrival at Saturn. Both will spend years orbiting and observing their respective planets. Some examples of differences in the distributed operations approach of these missions are shown in Table 1. The most significant difference is in instrument command verification by the central GS. MO has a list of validated commands that have been proven in system testing or in-flight checkout to be safe, predictable, and effective. MO investigators are free to use these command at will with the GS ensuring that the header is correct. Cassini GS personnel currently see their spacecraft as more subject to instrument driven pointing and power conflicts. They are weighing whether every sequence should be validated through full simulated expansion of the instrument commands to ensure that the instrument sequences do not adversely affect the total sequence or spacecraft. This concern also motivates the current plan to leave the instruments turned off for the first five years of cruise, until adequate ground software can

be built to operate them. Clearly, this baseline plan will improve in favor of more free and early instrument operation as the Cassini teams become more comfortable with distributed operations.

4. CONCLUSIONS

The distributed data system approach for Planetary Science Operations provides more flexibility and allows tradeoffs to be made between the various components of the whole system in order to optimize the end-to-end data return and reduce project costs. Translations of technical expertise to another implementing or operating agency are reduced. Mission inheritance and adaptability are enhanced.

MO is having excellent success in their distributed operations. The Cassini Ground System Review Board finds, "...this is the correct approach for a mission of this type which will fly in the late 1990's." Our challenge is to build on the distributed approach for more responsive, lower cost operations.

Acknowledgement

This work was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

SCOSII: ESA'S NEW GENERATION OF MISSION CONTROL SYSTEMS

N 94 - 23851

J. F. Kaufeler and N. C. Head

Flight Control Systems Department
European Space Operations Centre
Darmstadt, Germany

ABSTRACT

The paper describes the next generation Spacecraft Control System infrastructure (SCOSII) which is being developed at the Operations Centre (ESOC) of the European Space Agency (ESA). The objectives of the new system and selected areas of the proposed hardware and software approach are described.

Key Words: software, mission control, object oriented, infrastructure

and discusses the improvements over the earlier generation systems.

Two viewpoints are taken for these discussions, corresponding to the two major user communities; the mission software engineers who must configure and extend the infrastructure in order that it be usable for a specific mission and the end users, that is the operations engineers who must control the spacecraft with the help of configured software.

1. INTRODUCTION

The Operations Centre of the European Space Agency is currently developing the next generation of its Spacecraft Control and Operation System (SCOS).

Over the years ESOC has made great efforts to develop and use mission independent and configurable control system software. This software is also sometimes referred to as multi-mission software, in that it can be configured to support different missions. The first system of this nature was the Multi Satellite Support System (MSSS) which entered service in 1977 with the launch of GEOS-1; this was followed by the first SCOS systems which are used to support the Hipparcos, ERS-1 and Eureka missions. It will continue to be used for all launches until approximately 1995. Ref. 1 provides an overview of the history of ESOC spacecraft control infrastructure.

This paper describes some of the principle characteristics of the next generation system (SCOSII)

2. OBJECTIVES

2.1. Functional improvements

The end users require a number of improvements and extensions to the functions provided by the system in order to mitigate the increasing workload resulting from the complexity of modern spacecraft. Use of onboard microprocessors to provide some level of autonomous functions on the spacecraft is an example of such complexity.

The essence of any control system is the description of the device (in this case a spacecraft and its components) to be controlled and its expected and desired behaviours under various conditions. With growing complexity of spacecraft the expressive power of the database oriented description techniques used in previous generation systems is no longer adequate.

It is also clear that there is a need for improved performance and flexibility of the Man Machine Interface elements of the system; in particular inadequacies in the MMI have sometimes slowed down the response of operations engineers to

the contingency situation.

These lacks are seen to be of two types:

insufficient ability to obtain synthetic overviews of high level status of the spacecraft and reduce the information overload on the operations engineers

slow response when reviewing historical data in order to evaluate spacecraft performance or to locate the cause of anomalous events.

In view of these problems two of the major goals of the new infrastructure are to (1) improve both the functional scope and flexibility of the system and (2) provide greatly increased performance as seen at the man machine interface.

2.2. Maintainability

The software engineering community have voiced concerns about the maintainability and configurability of the existing systems; the effort involved in adapting these systems to the needs of a specific mission is significant, it has to be repeated (at least in part) for each update of the infrastructure which is both tedious and error prone.

Typical activities involved in configuring the infrastructure for use with a specific mission fall into two broad categories; specification and development of new control and analysis facilities in support of special characteristics of the target spacecraft which cannot be handled by the generic software (for example the control and monitoring of an Onboard Computer) and modifications to existing portions of the system to account for small differences in behaviour (for example different command encoding schemes or unusual telemetry layout policies).

The new infrastructure will apply some of the more recently developed software implementation approaches and technologies in an effort to reduce the effort involved in both development and maintenance of such specific mission configurations.

3. TECHNOLOGIES

There have been some recent advances in the areas of both spacecraft and software engineering which contribute to the concepts of the SCOSII system; with hindsight it is clear that previous versions of the infrastructure have been handicapped by the lack of these advances.

3.1. Standardisation

The application of standards in spacecraft design and operations, either explicit or *de facto*, will obviously greatly ease the provision of infrastructure software systems by helping to reduce differences between missions. The SCOSII development takes advantage of a number of such standards.

3.1.1 Explicit standards

The Consultative Committee for Space Data Standards (CCSDS) has produced definitions for the space/ground link protocols (Ref. 2) and transport data formats; although concerned mainly with the issues of data transport they will considerably reduce the proliferation of transport protocols and simplify the job of interfacing to the ground station equipment which (in ESA systems) provide the transport services.

ESA's Committee for Operations and EGSE Standardisation (COES) is working on a standard (Ref. 3) which will cover many presentation layer issues; this will prescribe the telemetry and telecommand data encoding schemes for the data contained within the CCSDS packets. It also defines the protocols to be applied for control of certain onboard applications (Master Timeline management or Onboard monitoring for example). These matters are of direct relevance to control systems which have the job of presenting the telemetry contents to the engineers and controlling the onboard applications. Despite its preliminary nature this standard is being adopted as one of the foundation stones of the SCOSII system.

3.1.2 de facto Standards

The European space industry is developing, in cooperation with ESA, a set of effectively standard components for the construction of space-

craft (typical "almost" standards are to be found in the onboard data handling systems as well as in low level components such as command decoders). As these devices are employed on more and more spacecraft it is planned that the relevant portions of the ground control systems will also be re-used.

ESOC has also published a set of guidelines (Ref. 4) for spacecraft manufacturers describing the operational constraints which should be respected by ESA spacecraft if they are to be operated by ESOC. This document is in the process of being transformed into a COES standard.

3.2. Object Orientation

The Object Oriented approach to analysis, design and implementation of software systems has now matured to the extent that it has been adopted for the implementation of SCOSII.

The intention is to make use of the inheritance and polymorphism properties of the technique to promote the use of specification (and implementation) by "difference". It is expected that such an approach will have two benefits; firstly, the differences between a proposed operational facility and an existing one will be explicitly stated and reviewed and will be subjected to more intense scrutiny than if they were simply part of an overall description of the facility and secondly, the implementation and maintenance of the modified facility will be limited to the code which provides the differences in functionality.

These benefits will however only be realised if a thorough analysis of the problem domain is applied to new mission systems and if the various users of the infrastructure are prepared to resist the temptation to "re-invent the wheel" for each mission.

4. ARCHITECTURE

4.1. Hardware environment

The SCOSII system will be implemented on a network of Unix workstations. This is a major departure from previous infrastructure systems which have been centred around a host machine with, in some cases, intelligent terminals for user interfaces.

This change is motivated by two factors; the availability of large amounts of extremely cheap computing power compared to the centralised approach and a desire to achieve a system which is more tolerant of failure of either processor hardware or software.

The initial platform selection for SCOSII consists of SUN Sparcstations connected by an Ethernet network. Each workstation will have local storage (at least 1Gbyte per station), 2 or 3 colour screens and pointing device (mouse or track ball). The processors will run the SUN SOLARIS 2.x system which is System VR4 compatible and provides a number of services making it more appropriate for real-time or time-critical usage than the previously available SUNOS versions which were derived from BSD Unix.

A typical mission configuration will have one workstation for each operations engineer position plus a further 2-3 server nodes providing common functions. It should be noted that there will not be a Network File System (NFS) server; this has been identified as a potential single point failure. Each workstation will use some portion of its local disk space for storage of executable images and static configuration data and will obtain dynamic data from one of the functional servers located on the network as needed. It is hoped that a minimum service can be identified which can run without any support from these servers and allowing each individual workstation to function independently to a limited extent.

An important goal of the architecture is to ensure that the physical location of system functions (in particular the servers) is transparent to other applications. This will allow a small system for a simple, low data rate spacecraft to be configured on a single workstation.

4.2. Basic Software

The underlying approach to the structuring of the SCOSII software is that of the client/server pair. Much of the functionality of the system will be provided by servers of various kinds (a telemetry server or an uplink server are possible examples). These will communicate with their clients over the network. In order to help keep the resulting network load within manageable limits the concept of a "broadcast" server has been introduced.

This concept makes use of the observed fact that, in a multi-user control system, most of the users perform roughly similar activities at roughly the same time and so will have very similar needs for server data. A broadcast server will therefore distribute its replies to all user nodes on the network with the assumption that it is, in most cases, directly relevant data and will avoid a subsequent repeat request for the data from each user workstation in turn.

As the broadcast data may not be immediately applicable when it arrives at a node (due to small differences of processing sequence for example) it will be stored in a cache which is maintained by each node in its virtual memory using the local disk storage as backing store. This cache is then checked before issuing possibly redundant requests to the server for the data.

The cache management policy applied at a particular node may be subject to tuning depending on the nature of the applications running on the node; an engineer performing evaluation of a manoeuvre which took place several days or months ago is unlikely to be interested in real-time telemetry broadcasts, similarly the active spacecraft controller workstation will probably be configured not to cache data older than a few days. These tuning activities are not visible to the broadcast server; these behave in the same manner at all times.

4.3. Basic system model

As noted earlier it is important for a control system to be able to model the behaviour the device being controlled to a sufficient level of detail. Previous systems have attempted to provide this flexibility by the use of a single "engine" driven by a set of data tables of predefined layout. The approach cannot, in practice, describe all kinds of spacecraft behaviour and has resulted in a large amount of mission specific software being produced to handle devices which are not describable in this way.

The SCOSII infrastructure adopts a different approach, suggested to the designers by the techniques applied in Object Oriented analysis and supported by the technology available in the chosen object-oriented programming language (C++).

A spacecraft is generally viewed by the engineers as an assembly of sub-systems (and sub-sub-systems etc). The behaviour of each component sub-system is generally fairly well defined as are its relationships to other sub-systems on the spacecraft. Thanks to the capabilities of Object Oriented software and databases this view of the target has been adopted as the foundation of the SCOSII "database"; each sub-system is represented as an object in the database. These objects maintain connections of various kinds (contains, is_part_of, refers_to etc) to the objects representing the other portions of the spacecraft with which the sub-system interacts. The benefit

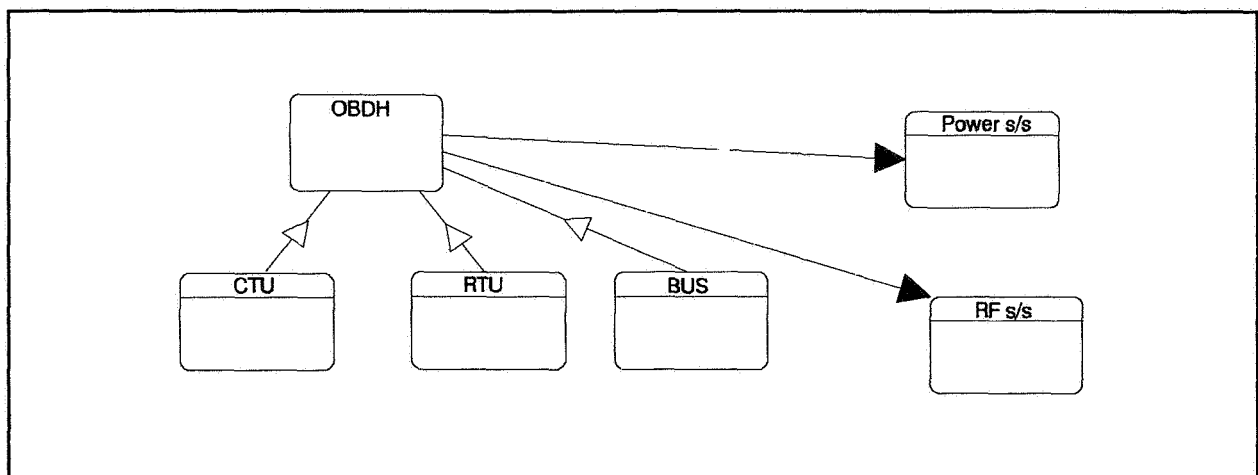


Figure 1 - Representation of an OBDH

of this becomes clearer when it is realised that each object may have a specific "engine" to define its behaviour and may use data structures best suited to the task of describing the object in question. A complete discussion of this topic is outside the scope of this paper but a brief example should help to explain the overall approach.

Figure 1 shows how a simple Onboard Data Handling (OBDH) sub-system might be represented in this manner. It contains several items (a Central Terminal unit, a number of Remote Terminal units, a data bus) and makes use of information from several other sub-systems (power, RF etc). This does not claim to be complete or fully correct. However it serves to demonstrate the principles involved.

The effort involved in setting up such models of a spacecraft will be large, even if sophisticated editors are provided, and it is necessary to take additional steps to help. As noted earlier, there is an advantage to be gained in specifying, and implementing, systems "by difference" from existing systems. This principle can also be applied to the modelling of the spacecraft. A basic set of foundation objects can be provided which can

then be specialised to match a specific need for a specific spacecraft. This technique is referred to as "inheritance" in the Object Oriented view. Again a full explanation of this is outside the scope of this paper but an example should serve to clarify some of the issues.

Figure 2 shows a hierarchy of objects, using a notation loosely based on that of Coad/Yourdan (Ref. 5) which represent an onboard processor of an imaginary mission - DemoSat. At the highest level is an object representing a generic onboard processor which has a number (at least 1) of memory banks and a watchdog timer. At the second level is a multitasking processor which *is_a* generic processor and therefore has all the characteristics of a generic processor but, in addition, has descriptions of a number of Tasks which are able to run on the processor. Finally comes the DemoSat processor which *is_a* multitasking processor. The control logic for the DemoSat processor is similarly derived; as it *is_a* generic processor, for example, it is able to perform memory loads to its memory bank(s) (typically open the memory bank, send a number of load packets, close the memory bank again) and similarly understand the manipulation of the watchdog timer. The multitasking processor component pro-

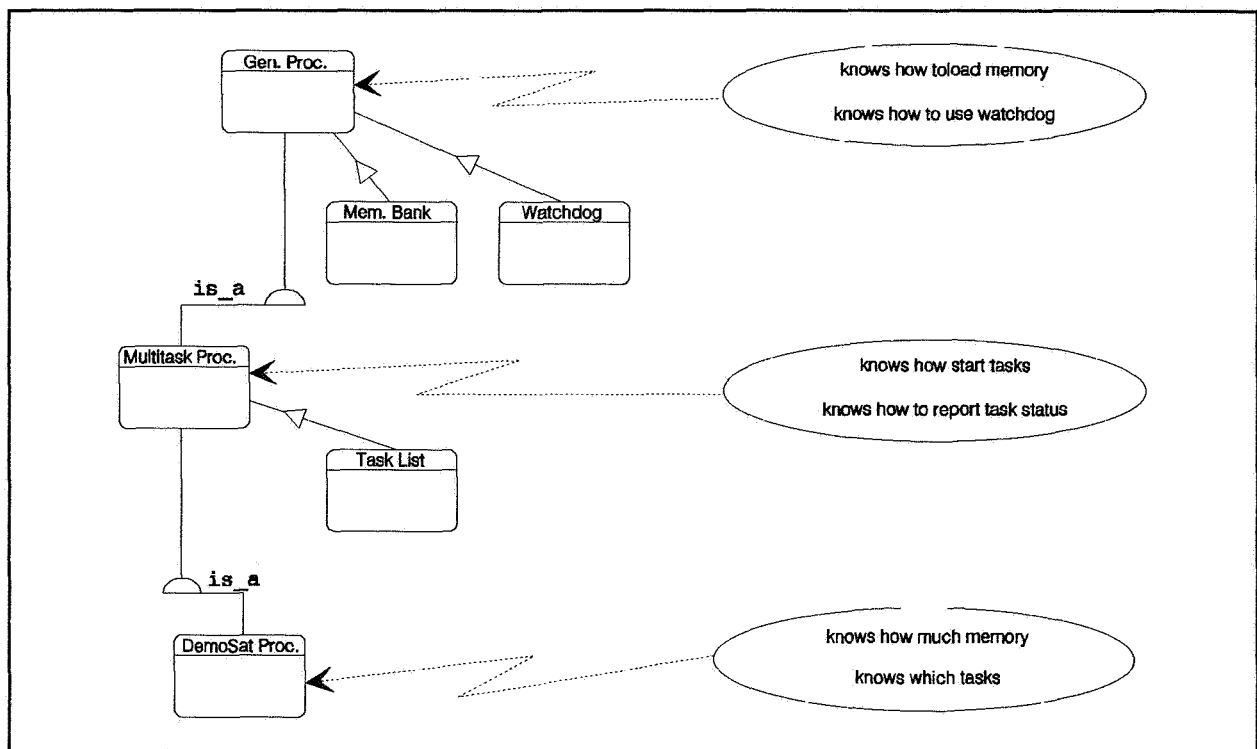


Figure 2 - Representation of a Processor

vides additional behaviours (for example start and stop of tasks, report on task status) and finally the DemoSat processor specific component provides information about the exact number of memory banks, the exact list of tasks which can be run, their constraints and so forth.

The major point of interest (and labour saving) is that the objects representing the generic processor and the multitasking processor have no mission specific elements and could be implemented once only; the implementation of the DemoSat processor object then requires only provision of those behaviours which are not covered by components or which are not standard (implementation by difference) and specification of particular characteristics (number of memory banks, valid task list etc) which are referred to but not defined by the higher level components.

It is clear that it will be necessary to perform a careful analysis of typical spacecraft in order to build a usable library of such generic objects which can be reused at some later date; the first clients of SCOSII may not reap all of these benefits and may actually be involved in slightly more work to participate in the analysis and creation of such standard building blocks.

4.3.1 Applications software

The actual applications tasks of SCOSII can be viewed mainly as gateways to the functions provided by the objects in the spacecraft model. To continue the example used above the Onboard Software Maintenance application will be mainly a shell providing a user interface to generate messages to the DemoSat object and providing an area of screen where the DemoSat processor will generate its visible representation.

Those parts of the representation which arise from the generic processor (i.e. watchdog status) or from the multitasking processor (i.e. task status list) will not need to be reimplemented. This helps reduce effort and, perhaps more importantly, ensures a consistent user interface from one mission to the next wherever there are similar functions to be performed.

User interfaces in the SCOSII system will be provided via the X.11 & OpenLook standards with specific objects (a CTU object for example)

providing the logic for processing input events and using a X-window provided to place their visible image. The image produced may vary depending on the context of the application; in some cases the current value will be needed, in other cases a "database" edit dialogue will be requested. The applications will be responsible for providing this context information.

5. SUMMARY

This paper has provided a brief glimpse of the plans for the next generation of the ESOC spacecraft control infrastructure - SCOSII.

It has discussed some of the motivations behind the decision to implement a new version of the infrastructure and has presented some of the areas where the project is planning to deviate from established practice in an attempt to improve the service offered both to the end users (the spacecraft engineers) and to the, oft forgotten, immediate users of infrastructure software who are responsible for the customisation and maintenance of the software for specific missions.

It remains to be seen whether all of the benefits hoped for by the SCOSII team will actually be realised in practice but initial positive responses have been received from all sides. We are taking a cautious approach however; the very first version of SCOSII will contain only functions known to the users from previous generations of infrastructure systems with introduction of the more advanced aspects being left until the safety net of backwards compatibility is complete.

6. REFERENCES

- 1 - Debatin,
SPACEOPS 1992
- 2 - CCSDS,
Packet TM & TC standards
- 3 - COES,
Packet Utilisation Standard
- 4 - SOIRD,
Spacecraft Operations Interface Requirements Document
- 5 - Coad/Yourdan,
Object Oriented Analysis

Operations Concepts for Mars Missions with Multiple Mobile Spacecraft

N94-23852

William C. Dias

California Institute of Technology
Jet Propulsion Laboratory
Pasadena, CA USA

ABSTRACT

Missions are being proposed which involve landing a varying number (anywhere from one to 24) of small mobile spacecraft on Mars. Mission proposals include sample returns, in situ geochemistry and geology, and instrument deployment functions. This paper discusses changes needed in traditional space operations methods for support of rover operations. Relevant differences include more frequent commanding, higher risk acceptance, streamlined procedures, and reliance on additional spacecraft autonomy, advanced fault protection, and prenegotiated decisions. New methods are especially important for missions with several Mars rovers operating concurrently against time limits. This paper also discusses likely mission design limits imposed by operations constraints.

Key Words: Rover missions, planetary missions, Mars missions, autonomous spacecraft operations, sample return missions, surface operations, round trip light time

1. INTRODUCTION - THE PROBLEM

The most familiar planetary operations techniques are oriented towards planetary encounters / flybys, long-duration orbital missions, or stationary surface landers (Ref. 1). While similar in some respects to stationary surface landers, a rover's mobility introduces important new operations requirements.

The essential differences between rover operations and immobile surface operations boil down to two: (a) round trip light time delay combined with a non-deterministic

spacecraft environment, and (b) ground decision making speed. These are introduced briefly below. It is important that BOTH problems must be addressed. Solving only one of the two does not bring about a satisfactory rover mission.

1.1 Light Time Delay

Although JPL operations long ago learned to compensate for lengthy time delays introduced by round trip light time, the techniques which have evolved to support these kinds of missions are not sufficient for command and control of rovers, especially rovers being operated against time limits. This is because whereas stationary landers can be commanded as if their spacecraft states were deterministic, rover state changes more frequently and less predictably. The rover encounters and must react to obstacles whose exact effect cannot be predicted with much certainty beyond a few feet ahead, if then. A rover commanded in the traditional way would have to stop, downlink, and wait for recommanding every few feet. Since the light time delay to Mars is up to 22 minutes (and since many low energy trajectories from Earth to Mars typically arrive near the time of maximum light time delay! (Ref. 2)), rover progress would be unacceptably slow for many desirable mission scenarios.

As will be discussed below, the light time delay problem will be solved by onboard autonomy and changes to conventional spacecraft fault protection strategies. However, applying these solutions without speeding up ground decision making, as discussed in the next paragraph, could tend to result in rovers with too much autonomy.

1.2 Ground Decision Making Speed

Many spacecraft operations decisions are made by committee. Reasons include the fact that some decisions are too much responsibility for one person, the need to satisfy a diverse customer community, and the distributed nature of the engineering knowledge base.

Without speeding up ground decision making, autonomy would tend to result in rovers which either spend a disproportionate amount of time waiting for earth redirection, or they would have to be programmed to fulfill too many objectives without checking with ground for new priorities.

Sometimes it is proposed to get rid of committees for the sake of quicker decision making turnaround for rover missions. It is argued they could be replaced by a single well-trained operator assisted by automation. We take the view that there are too many diverse customer communities for this to come about, and until many rovers have been to the planets, so much responsibility cannot be vested in a single individual. We must streamline the decision making, true, but all legitimate customer communities must feel they are represented in the decision making process. First, we propose to have small negotiating committees, empowered to make decisions for the whole community. Second, we propose that the decision making process be guided by a "decision tree", prenegotiated between the customers and the project before the surface operations phase.

2. COMPOUNDING THE PROBLEM

2.1 Multiple Rovers

Missions with more than two rovers provide different operations problems from those with only one or two. The desire to reduce mission risk results in plans to provide for multiple rovers per mission: if a few fail, the mission can still be considered a success. But there's a downside. The operations system has to be able to operate a larger number of elements concurrently.

This tends to increase the number of personnel and interfaces, and magnify expectations of mission return.

2.2 Time Limits

Spacecraft operations are usually up against time limits. Regardless of the type of mission, funding is set to stop at a given time. Spacecraft consumables and mechanism lifetimes may be running out. Rover missions can have additional time limits.

If multiple rovers are used serially, in an operations environment with a minimum number of personnel, there will be a lot of pressure in conflicting directions to: (a) finish one rover mission quickly, or park it, and start the next one, (b) go slowly on the first mission to get the maximum out of it, wearing out the first rover before the second is deployed, and (c) switch to parallel and coordinated operations to speed up mission return.

Conversely, when multiple rovers are run in parallel, again with small operations teams, one can expect many more deadlines per day. Attempts to operate without increasing the number of personnel will result in increased risk for individual rovers.

Sample return missions have an additional source of time pressure. Samples must be returned to the lander in time for final sample packaging and return rocket ascent to meet low energy return flight schedules.

3. SOLUTIONS

This paragraph discusses the range of solutions proposed to mitigate the above problems for small-rover missions being most actively proposed in NASA at the present time.

3.1 Onboard Autonomy

Onboard autonomy is needed so that the rover can continue to work without being recommanded more than once per day. It must be able to react to many unexpected

changes in its environment as it moves along a planned course. Our studies indicate that a Mars rover mission without autonomy might be expected to have a disappointingly low mission return (Ref. 3).

Autonomy must come with a specialized, higher-order spacecraft command language. This should serve to effectively eliminate "sequencing" in the usual sense from the routine operations tasks (though some sort of software simulation could still be utilized). In effect, the lower-level sequencing and command development functions, conventionally done on the ground, are moved onboard. The use of this limited autonomy means that the rover schedule is non-deterministic. (Note that low-level commanding and deterministic control is still available to ground for exception conditions: see paragraph 3.7.)

JPL has developed and proposed a "behavior-based" method of rover command and control (Refs. 4, 5). It attempts to achieve high level objectives provided by earth commanding, while reacting to obstacles within the range of its programming. Although not yet fully developed, the basic capability has been demonstrated (Refs. 5, 6). This is opposed to forms of autonomy proposed in earlier years for large-rover missions, in which autonomous rovers perform onboard replanning (Ref. 3). For that reason, the onboard decision-making architecture is quite different from that proposed for those larger rovers (Ref. 7).

It is important that this form of autonomy is used strictly to simplify the mission operations and enhance return. Complex, subsystem-oriented autonomy is sometimes proposed for other spacecraft which has the ironic effect of increasing the number of engineering activities and the need for earth checking of spacecraft states. Autonomy with these effects would increase our costs while reducing mission return and should not be considered.

3.2 Maintain Fluid Expectations

Granting autonomy to the spacecraft results in a non-deterministic situation. Exact simulations of mission results cannot be expected. For example, if 10 activities are programmed for a certain day, the rover might execute only six if it finds the environment out of the expected range at some point. Alternately, it could execute more than 10 activities if programmed to attempt autonomous recoveries. All this means that even data quantities cannot be projected exactly with respect to time.

Simulations of rover activity may not be accurate, but it is important that people do not conclude that a lack of accuracy here is any kind of a mission failure. This could present a problem to those who feel a need for these exact projections.

3.3 Reduced Team Sizes, Simplified Procedures

A core operations team for each rover would be reduced in size, to be able to turn around uplink commands at the rate of one a day. This small team would need to organize its activities around the Mars day length for certain periods, depending on mission duration and objectives. Time would still be needed for daily meetings with other rover teams and management. This keeps the core team in touch with the rest of the project and the outside world. It is needed because of continually changing priorities, the need to keep each other abreast of newly discovered facts, and (we must assume) spacecraft problems. The use of prenegotiated decisions (see 3.5), will facilitate streamlined procedures.

3.4 Daily Command Turnaround

We have proposed daily ground recommanding of Mars rovers. After trial and error with various scenarios, it turns out that daily turnaround seems to provide for a reasonable amount of rover activity, a reasonable amount of earth control with frequent opportunities for redirection, and enough time to receive data describing each daily activity period and spacecraft health.

An exception allowing for two command turnarounds per day is being considered for specific, highly constrained situations in missions where daylight periods are long and projected surface lifetime is short. These fast command turnaround scenarios will be facilitated by prenegotiating decisions where possible, paragraph 3.5.

It is possible to conceive of highly autonomous missions where rovers simply wander around and send back results. Recommending might be sporadic, and would be devoted to resetting the priorities of the autonomous control system as the mission progressed. None of the Mars rover missions currently being proposed seem to fit this type of scheme.

3.5 Prenegotiated Decision Tree

In order to speed up ground decision making, we foresee the need to negotiate decisions before landing. The results of this negotiation will be a decision tree in which we try to anticipate as many situations as reasonable. An attempt will be made to "cover the waterfront" so that when situations don't meet the prenegotiated ones exactly, there are at least enough similarities to speed things up. We are guessing this decision tree might have on the order of 100 nodes, depending of course on the type and scope of the mission.

3.6 Rover Fault Protection Strategy

JPL spacecraft are normally equipped with "fault protection" capabilities which are automatically invoked whenever onboard diagnostics detect error states. These routines initiate a canned recovery procedure. Generally the spacecraft is forced into a safe state, but one that is nonproductive with respect to mission return, until returned by the ground to a productive state. Mobile spacecraft autonomy calls for a change in the usual concept of spacecraft fault protection. This is because the rover frequently gets into unpredictable states during autonomous operations. If the usual fault protection were allowed to interrupt the rover's progress at those times, we would get no

benefit from the autonomy. Luckily this change need not take the form of a wholesale redesign, and need not engender undue acceptance of new risks.

Assuming NASA Class A (ie, low-risk) rover missions, we still need fault protection against pathological conditions. What needs to be added is a layer of protection between behavior control and fault protection. This layer handles conditions which are in some sense unexpected but not pathological. This layer corresponds neatly to what roboticists refer to as "reflex" technology. The rover uses reflexes as part of its autonomy to get through a large number of unexpected but generically foreseen obstacle conditions, while staying roughly on course, but is programmed to invoke the lower levels of fault protection when these reflexes fail to keep it out of trouble. Whereas the behavior control layer and reflexes are non-deterministic, the fault protection modes below the reflex level are deterministic, to facilitate troubleshooting and recovery.

Missions below NASA Class A can take more risks as part of lowering mission costs. The rover missions being most actively considered at this time fall into these classes. Fault protection may be less rigorous, perhaps offset by element-level redundancy, ie, more rovers per mission. In these cases, robotic reflexes will still be used as part of a natural implementation of the autonomy scheme.

3.7 Lower Level Commanding

Unless it is decided that higher risk is acceptable for some rover mission, all the traditional spacecraft and operations functions need to be available for troubleshooting in the event of spacecraft problems. Fallback capabilities must include the ability to override the autonomy and use a set of low-autonomy spacecraft operations commands and modes. These low-autonomy modes correspond to conventional spacecraft operational modes. Unlike the non-deterministic operations modes discussed above in connection with autonomous operations, these modes are

deterministic. They may be used to get out of fault protection states, or to operate in environmental or hardware failure situations for which the autonomy is not programmed. In general, operating extensively through low-level commands must be expected to greatly reduce mission return.

3.8 Serial Operations Preferred

Relying on parallel operation of multiple rovers should be avoided until we have a better hands-on understanding of rover mission pitfalls -- most obviously uncertainties about the terrain. It should be recognized, though, that it may be desirable to include more than one rover even on near-term missions, to reduce the risk arising from depending on just one unit. In the event that several rovers are landed close in time, it is proposed that rovers nevertheless be deployed and operated serially through the point of mission success for each rover, then reactivated and selectively operated in parallel to increase mission return.

4. RELATIONSHIP TO PROPOSED MISSION SET

This paragraph relates the above principles to three proposed rover missions: MESUR Pathfinder, MESUR Network, and a small-rover Mars sample return mission.

MESUR Pathfinder may be authorized as a precursor to MESUR Network. It is presently conceived as having one or two landers. One rover per lander may or may not be included in the flight system. If two rovers and landers are flown, and if the two aircraft are provided delta-V for separated arrival dates, serial operation of the two rovers is envisioned. If a short surface mission duration is decided on, and the lander(s) is targeted for an area with a long daylight period, two command turnarounds per day may be attempted.

MESUR Network is presently conceived as having 16 landers, some or all of which may have a rover. We expect the landings in waves of four or eight, possibly separated

by a few weeks, possibly separated by the two years between launch opportunities. An orbiter assists in communication, but not rover navigation. We picture deploying and operating each rover serially for about ten days. Once each rover from a given landing "wave" has been deployed and operated for ten days, we foresee switching to parallel operation of selected rovers for enhanced mission return.

A small Mars sample return mission has been described which has two landers and one or two rovers per lander (Ref. 8). In this case an orbiter is needed for communications and assistance in navigation. Rovers would be asked to cover up to several kilometers in the course of a mission, to get a diversity of samples. A single command might be sufficient for a kilometer traverse. The current thinking is to run all the rovers in parallel, once one sample has been stored in each lander using serial operations. A separate operations team is needed for each rover, and their activities must be timed with respect to local daylight to optimize return.

5. ACKNOWLEDGEMENT

The work described in this paper was carried out at the Jet Propulsion Laboratory / California Institute of Technology under a contract with the National Aeronautics and Space Administration.

6. REFERENCES

1. Linick, T. 1985. Spacecraft commanding for unmanned planetary missions: The uplink process. Journal of the British Interplanetary Society, vol 38, no.10, 450-457.
2. Bourke, R., et. al. 1989. Design of a Mars Rover and Sample Return Mission. Proc. of CNES International Symposium on Space Dynamics, Nov 1989. Toulouse, France.

3. Pivrotto, D. and Dias, W. 1990. United States Planetary Rover Status - 1989 (Report 90-6). Jet Propulsion Laboratory, Pasadena, CA.
4. Desai, R. and Miller, D. 1992. A Simple Reactive Architecture for Robust Robots. Proc. of IEEE Conference on Robotics and Automation - Workshop on Intelligent Architectures for Autonomous Robots. Nice, France, May 1992.
5. Miller, D., et. al, 1992. Reactive Navigation through Rough Terrain: Experimental Results. Proc. of AAAI 1992 National Conference on Artificial Intelligence, pp. 8.3-8.8. San Jose, CA, July 1992.
6. Rover demo. 1992. "Rocky IV". Videotape, June 26, 1992. JPL File No. AVC-92-145. Jet Propulsion Laboratory, Pasadena CA.
7. Dias, W. C., and Zimmerman, B. A. 1990. System Control of an Autonomous Planetary Mobile Spacecraft. In Telematics and Informatics, vol 7, nos 3/4, 323-339.
8. Sturms, F. (ed.). 1991. Concepts for a Small Mars Rover Sample Return Mission Using Microtechnology. (JPL Internal Report D-8822) October 1991. Jet Propulsion Laboratory, Pasadena, CA.

PREPARING THE DSN OPERATIONS CONCEPTS

N94-23853

A. J. Salazar, M. D. Bailey, N. R. Kuo, and B. M. Wilkinson

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California, USA

ABSTRACT

Operations concepts are prepared to support a specific application and can, therefore, be classified accordingly. Studies of NASA and military operations concepts suggest three major types: data services, customer services, and systems oriented. Data services concepts concentrate on data types (e.g., telemetry and command) and how these data are processed and delivered to the user. Such concepts are normally used by data-processing centers to describe data-coding schemes and data formats. Customer services concepts concentrate on the customers' requirements and describe how these requirements are met using the systems available to the operation. Project centers use such concepts to describe the various types of data as they flow from the source (e.g., a spacecraft) to the end user (i.e., the customer). Systems oriented concepts concentrate on the use of systems for processing and delivering customer data. This paper examines the latter type and the "concepts" that are inherent within them.

Key Words: Network operations, system requirements

1. THE ERA OF APPLICABILITY

The era of applicability for any operations concept is directly related to its purpose. If the purpose of an operations concept is to describe operations in the present environment, then its era of applicability is now, until the environment changes. But if that purpose is to guide the system development activities and establish the amount of operational resources required to operate the network, then its era of applicability is some time in the future.

1.1 Operations in the Future

System oriented concepts are better suited for environments that change over time, in preconceived ways. The Deep Space Network (DSN) is such an environment, and its operations concept was chosen so that its system design plans satisfy the primary requirement of maximizing services to DSN customers.

1.2 Selecting the Era

To choose the era of applicability, one must know the design plans for the network. For example, the DSN plans call for a slow evolution from extremely complex operations to increasingly automated systems that will be easier to operate. The changes are planned to occur over the next ten years, and thus the era of the operations concept was chosen to be midway in that evolution. About five years hence, operations of the network will be different from today's, yet not so different that today's operations cannot be recognized. If the plans hold, it should not be difficult to derive from the operations concepts the amount of operational resources required to operate the network.

2. BEGIN WITH THE BASICS

An operations concept, whether prepared for a data-processing center, project operations, or a tracking network, must begin with a definition of the basic purpose for which the entity was created. This must be done first in order to build the foundation of the operations.

2.1 An Example — the DSN

The basic definition of a tracking station's purpose is that a spacecraft tracking network

exists for network customers to communicate with their spacecraft. In other words, the network is the vehicle for obtaining data from a spacecraft, for sending data to a spacecraft, and for providing tracking data to help the customer navigate the spacecraft. The network must therefore provide the means for accomplishing these tasks, always striving for one-hundred percent customer satisfaction. The relation of this operations requirement to the design of network systems is implicit.

2.2 Building the Foundation

Before constructing an outline, the most critical operation (i.e., the foundation of the operations concept) must be defined. The best way to do this is to begin with the basics and chart out the process from beginning to end. For example, in the DSN data are acquired, preprocessed, and delivered to the project centers or the spacecraft. The spacecraft data must be acquired for the other processes to exist. Therefore, one can emphatically state that data acquisition (including radio frequency signal acquisition) is the most critical operation. The foundation for the DSN operations concept then is data acquisition, as performed by the tracking antennas and front-end systems at the tracking complexes.

2.3 Using the Foundation

Having identified the critical operation, the next step should be to describe this operation. This should be fairly simple for operations concepts that describe operations of current systems. The task is more difficult for operations concepts that describe operations of partially implemented and future systems. Obviously, this will take a considerable amount of time and coordination. Other concepts, described below, must be also be considered, and if necessary, solidified.

3. OTHER CONCEPTS

3.1 The Congruency Factor

The DSN continues to do everything possible to satisfy the end customer. In the past,

this objective was accomplished by designing systems that provided the required technical capabilities with little regard for what it took to operate them. System operability was considered, but these capabilities were, as a rule, determined by the large amount of operations resources available at that time and the simplicity of the systems. However, as systems requirements became more stringent, network systems became more technically complex and thus more difficult to operate. It is understood that systems of this type will always have flaws, regardless of how careful the designers might have been. Experience also shows that the more complex the systems are, the more complex the flaws are, and the harder it is to develop "workarounds" or to fix them.

3.2 New Strategies

Recent directions are stimulating questions such as, How will the system designers ensure that the impact of any flaws is minimized? How will the designers ensure that data return to the end customer is maximized? This is where investment in an operations concept of the type described here will pay large dividends later. Knowing how the system will be operated in order to fulfill the basic purposes for which the center or network was created is another form of stating operational, and hence customer, requirements.

3.3 On Systems Capabilities

The operations concept should not have to acknowledge design flaws and deficiencies and allocate resources accordingly to operate, maintain, and sustain the network around them. Clearly, this is a waste of budgets and resources. Should the operations concept therefore "define" the type of systems that will reduce operational costs and increase data return to network customers? We contend that describing "workarounds" to account for system deficiencies is simply a way of describing system deficiencies. Thus, the rules for designing systems that operations may use must first be changed for the operations concepts to have any effect.

3.4 The Concept of Service

Businesses have shown us that the ability to make strict definitions is the direct result of making right or wrong assumptions. These assumptions lead to definite commitments by management as to where, when, and how much of a company's resources are to be deployed to increase sales, and thus are the key to profitability. In technical applications, such as the DSN, profitability is not a motive, but the deployment and use of resources to provide the best possible service to its customers is of paramount importance to network businesses. The operations concept, like its counterpart, the operations plan in a business, must identify how resources will be utilized to increase network (i.e., product) availability without increased costs. This means that it is necessary to define concepts that best fit the "industry" and to demand that systems be made more capable, robust, and operable.

3.5 Using the Service Concepts

The term "service" is often used in various contexts to describe the operations of a center, or network. Defining service is key to the way in which an operations concept will be prepared. One may begin by establishing a distinction between "data services" and "network services." Data services include telemetry data service and command data service. Network services, on the other hand, refer to the data services modified by the "goodness" of the resources that provide the data services. It could be argued that the data services encompass network services, but that depends on one's perspective. That is, "data services" (of the network) are most likely defined from the customer's perspective, while "network services" are defined from the perspective of network operators and customers alike. A tracking network's operations concept comprises more than data flowing from A to B—it accounts for the hardware, the software, the procedures, and the people who satisfy the requirements of network customers—a strict definition.

3.6 The Meaning of Transparency

With the present technology, it is possible for a spacecraft user (an experimenter) to sit at home and utilize command data services to send command data to his or her instrument on board a given spacecraft. He or she may also utilize telemetry data services to receive data from the instrument. In this case, the user does not have to worry about transmission protocols, nor concern himself or herself with how the data get to the spacecraft and back. So, as far as the user is concerned, the network is transparent. However, for the Project Operations Control Center (POCC) and the tracking network, such is not the case. Realistically, the user's command and spacecraft data (for spacecraft safety and security reasons) are preprocessed by the POCC and the network. The handshake between the POCC and the network is an established interface—strict and unforgiving. Therefore, from the operations concept standpoint, the network is not transparent, and the interfaces, as well as all the factors affecting these interfaces, must be allocated within the overall cost of operations.

3.7 The Interfaces

The operations concept must address all interfaces and describe how they will be operated in a particular era. The description should include the fact that the funnel (i.e., the network) can choke any data passing through it. That is the best way to illustrate that the network is transparent only when all data are delivered to the customer. When data are lost, the network is no longer transparent. The operations concept must include such facts as well as the desired recovery activities that will minimize future data loss.

3.8 On Data Latency

Does it make sense for a network to record the data and play it back to the customer at a later time, or to transmit the data to the customer as soon as it arrives in the network (i.e., in real time)? If it is indeed true that the

higher the latency, the higher the cost, then what is the latency, or cost trade-off, that our network can afford? The job of our operations concept writers is by no means easy—the answers to questions like this are constrained by system capabilities and management direction. The operations concept must address these types of issues because to be cost-effective is to deliver products to the customer quickly and economically.

4. BUILDING THE FRAMEWORK

4.1 Extensible Facilities

Extensible facilities complement the basic operation. For example, in the DSN these facilities are the Ground Communications Facilities (GCF) and the Network Operations Control Center (NOCC). Any support facilities and their functions should also be described. In the DSN, the extensible facilities include the DSN Logistics Facility (DLF), the Complex Maintenance Facilities (CMF), and the Depot Maintenance Center (DMC). Each of these facilities plays a significant role in all network activities. Without them, the basic operation would not be able to meet customer requirements.

4.2 Facility Interfaces

The operations concept must describe the internal and external interfaces for all the on-line facilities. The description should not be at the bit level; other documents exist that provide these details. Rather, the description should concentrate on how the interfaces are used and the products that these interfaces support. The aim is to provide system designers with a basic knowledge of operations to make network designs more effective and less costly.

4.3 Network Availability

Network systems availability is key to the quantity, quality, and timeliness of data delivered to network customers. Of course, the goal is 100 percent, but this figure is reduced by the negative characteristics of the systems' performance-modifying factors: operability, reliability, and repairability. Availa-

bility can be increased by the use of redundant systems, but these systems do not improve operability, reliability, or repairability.

4.3.1 Reliability

The operations concept cannot contribute much toward highly reliable systems designs. No matter how reliable a system is, someday it will fail. Providing data to designers after the fact is, however, a significant operations role which should be addressed in these types of operations concepts.

4.3.2 Operability

More can be said about highly operable systems. Descriptions of reduced complexity in the human interface of the systems should be included. In the DSN, the trend is toward high-level operator commands, high-level displays, and manageable lower level displays. Thus, the activities described in the DSN operations concept assumes that these capabilities already exist and are guiding the design of operable systems.

4.3.3 Repairability

A true measure of repairability is the mean-time to repair (MTTR). This measure of MTTR is a figure of merit that indicates how well systems maintainers were trained on systems' maintenance and operations. It also indicates how well the systems were spared and the quality of the tools provided for system repair. More often than not for new designs, the operations cost is assumed to be the same as that for the operation of current facilities. Thus, if properly prepared and used, the operations concept can play a significant role in the allocation of design costs. Typically systems maintenance requires a large operations budget. Maintenance costs can be minimized with a sound maintenance philosophy. Therefore, we suggest the use of an operations concept to promote this philosophy at a high level. The operations concept must candidly describe how maintenance of these systems will be performed and what spares and maintenance tools must

be provided for the maintenance program to be effective.

4.4 On Customer Requirements

In the DSN, the implementation of customer requirements is based on the ability to satisfy the technical and operational requirements. The operations concept must distinguish between these requirements clearly and concisely. Operations play a role analogous to frontline salespeople in business, relaying customer requirements to product designers. The thought here is that ultimately, operations (the salespeople) will be blamed for the quality of the product, and not development (the product designers).

5. CONCLUSION

We have described the types of operations concepts, provided a definition for each one,

and then attempted to document the concepts that should be considered with the third type, the systems oriented operations concept as used in the DSN. We have provided concepts that could be useful in preparing operations concepts of this type. We have used the DSN example, defined its basic operation as the foundation, and then followed through by completing the framework. By sharing our experiences, we hope to have outlined some of the pitfalls and problems associated with this task. Whatever type of concept is used, an operations concept is by no means easy to prepare—the support of management is essential.

6. ACKNOWLEDGMENT

The research described in this paper was performed by the Jet Propulsion Laboratory, under a contract with the National Aeronautics and Space Administration.

N94-23854

NASDA SATELLITE MISSION OPERATION SYSTEM AND OPERATIONS

Kousaku Yamaya, Fumiyoshi Yoshida, Harushige Noguchi,
Tetsuo Takahashi and Kazuhiro Tanaka

National Space Development Agency of Japan
Tsukuba, JAPAN

ABSTRACT

NASDA has recently developed a new tracking and control system as a basis of for future satellite mission operation. It is named type-I Space Operations and Data Systems (type-I SODS). The software of this system is separated into three parts: 1)operation and control system, 2)network system, 3)support and information system. The operation control system treats telemetry and command operations. The network system controls the communication line and ground station equipments to connect the satellite and the operation control system. The support and information system provides to other systems necessary information. JERS-1 which was launched in february of this year is the first satellite operated by type-I SODS. We explain the architecture and operation methods of this system using JERS-1 mission operations.

Key Words:satellite operation,
network, planning system

1. INTRODUCTION

NASDA has launched many types of satellites since 1975. For operating these satellites, we have developed many equipments and software. Meanwhile, after 1990's a satellite which requests very complex operation is forecasted. To operate this complex satellite efficiently, it is needed to develop a new system ,to adjust systems which already exist and to integrate these systems. As the first

step of a new system, NASDA developed a ground network operation system. This is called type-I SODS. This paper introduces this system and its operations.

2. SYSTEM ARCHITECTURE OF TYPE-I SODS

Type-I SODS consists of many hardware and software. In the first place, this paper explains the outline of these components.

2.1 Hardware elements

NASDA has four s-band ground tracking stations and one satellite operation center. Three of four stations are in JAPAN and each station has two s-band facilities. One station is in SWEDEN and it has one s-band facility. All facilities consists of antenna, transmitter, receiver, range and range rate equipment, base band equipment, data transmission computer and station control computer. Functions and performance of these facilities are almost standardized except the antenna and the base band equipment. The differences among antenna equipments are acquisition antenna existence and the antenna mounting type. The difference among base band equipments is whether the base band equipment has the function of keeping stored telemetry or not.

The satellite operation center is in Tsukuba, called Tracking And Control Center (TACC). There are line exchange equipment, main frame computer, several mini-computers and workstations and many personal

computers serving as terminals of these computers.

2.2 Software

The software of SODS is classified into three categories. The first is the operation and control system. The second is the network system. The third is the support and information system. The component of these systems is explained below.

2.2.1 The operation and control system

This system is responsible for the satellite telemetry and command operations. It is composed of Satellite Operation and Control System(SOCS), Satellite Telemetry and Operated Command Keeping System(STOCK) and Satellite dataBASE(SBASE). The function of SOCS is to send commands and to display telemetries. STOCK is a batch system to handle telemetry and command history. SBASE manages satellite information about command and telemetry.

2.2.2 The network system

The network system controls ground station equipments and line exchange equipment to connect a satellite on orbit and the operation and control system in TACC. This system consists of Tracking and control Station system(TS), sTation network control system(TC), communication Line Control system(LC) and Network Management system(NM). The TS is installed in the station control computer of ground stations and controls ground station equipments. The LC controls line exchange equipment. The TC is a front-end system of TACC for station control. The NM supervises network system over all.

There is one more software which belongs to the network system. It is gateway system(GW), which is responsible for connecting NASDA's

network and foreign agency's network. In this paper, the detailed explain of GW is omitted.

2.2.3 The support and information system

This system provides necessary information to other systems. It consists of flight dynamics system(FDS), Network Planning system(NP) and Satellite operation Planning system(SP). FDS has functions to determine satellite orbits, to plan orbit maneuver, and to calculate prediction. NP schedules network operations and generates network plans. SP is responsible for accepting satellite operation requests, making command sequence and sending it to SOCS.

3. THE OPERATION AND CONTROL SYSTEM AND ITS OPERATIONS

The data interface between SOCS, STOCK and SBASE is shown in Fig.1.

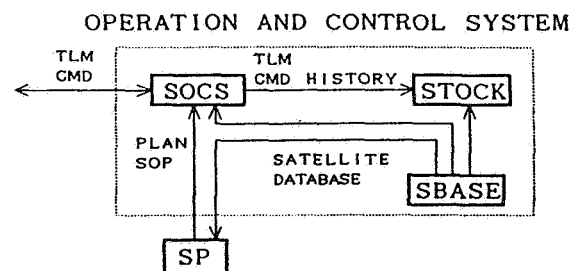


Fig.1 Data interface of the operation and control system

SOCS is developed for each satellite mission project. It is installed in two or three mini-computers to guard against computer failure. NASDA has now 5 mini-computers for SOCS. Ordinarily we use one mini-computer for one satellite operation, but in case of simple satellites, we use one computer for 2 satellites' operations. Thus we keep the operational flexibility.

To understand the SODS operation, the

command setting method of SOCS must be explained. SOCS has three methods for command setting. The first is the manual method. The operator types commands to send by SOCS editor. The second method is SOP calling. SOP is made by SP, and it consists of sets of commands and telemetries which must be verified after command sending. In case of the SOP calling method, SOCS displays the telemetries defined in SOP at the moment of command sending. But, if a command in SOP has parameters (for example time, magnitude value), the operator must determine it before command sending. The third method is calling PLAN. PLAN is also made by SP. PLAN consists of commands and SOPs of which parameters are already determined by SP. When the operator sets commands by PLAN calling, command editing is not needed.

Since, in the case of JERS-1, the operation is very complex, an operator uses the PLAN calling method. The PLAN is called 10 minutes before the operation start. In the case of a simple satellite, SOP calling method is usually used for command setting.

STOCK receives telemetry and command operation history from SOCS every day. STOCK has functions of keeping telemetry and command history, drawing telemetry graph, and printing the kept

data. STOCK is used for the satellite analysis.

SBASE is used to manage satellite database. It is mainly telemetry and command information, for example, telemetry word assign information, information to change from digital data to analog value, command name table, command verify information and so on. Satellite database is used by SOCS, STOCK and SP. SBASE has a function to notify the change information to other system.

4. THE NETWORK SYSTEM AND ITS OPERATION

This system operates according to the network plan which is made by the network planning system. In this section, we explain the action of network system as a whole and do not explain the individual software function. The interface of NM, TC, LC and TS is indicated in Fig.2.

4.1 Contents of a network plan

There are three kinds of network plans. The first plan is the normal network plan. It is a plan for satellite operation. The second plan is the stored telemetry transmission plan. This plan is used for sending stored telemetry which have been kept in base band equipment. The third plan is a dummy plan used to secure ground

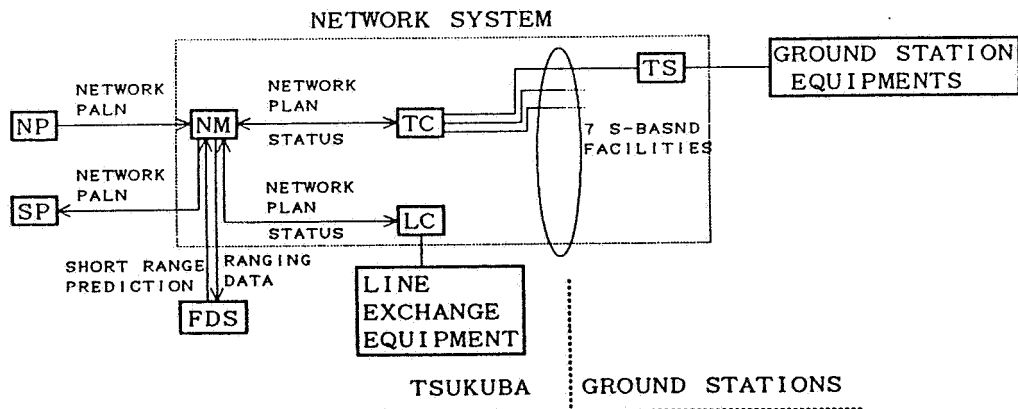


Fig.2 Data interface of the network system

station or SOCS computer from other network request. The last plan is used in case of hardware maintenance, development test and so on. A network plan mainly consists of the following elements.

- a) start time and end time
- b) satellite name
- c) ground station name
- d) SOCS computer name
- e) network operation

A network operation is a combination of these five elements: 1) telemetry, 2) command, 3) stored telemetry, 4) ranging, 5) 1-way doppler. In the case of a geostational satellite, the period of command, ranging or 1-way doppler is added to the network plan.

4.2 Network plan delivery

A network plan is made by NP. After NP makes network plans, NP send them immediately to NM. On the other hand, NM takes new prediction from flight dynamics system. NM always searches network plans, and if it finds the network plan which starts within 2 days, it appends the prediction to the network plan and sends it to LC and TC. When a network plan is sent to TC, TC sends it to TS without delay. Thus a network plan is sent and kept in TS and LC.

4.3 Equipment control of LC

The network plan is automatically carried out at the start time. LC controls line exchange equipment so that specified ground station and SOCS computer is connected.

4.4 Equipment control of TS

TS also starts to control equipments. In the case of normal network plan, the sequence of control is as follows.

- 1) control mode check phase
- Equipments of ground station have two

control mode. One is the local control mode and the other is the program controlled mode. It is selected by the station operator. When a network plan starts, TS confirms whether the control mode is program or not. If the mode is wrong, TS alarms the station operator by the recorded voice.

- 2) readiness check phase

TS sets equipment to test configuration and checks these functions: antenna motion, telemetry receiving and demodulating, command sending, ranging data acquisition and 1 way doppler data acquisition. If any anomaly is found, TS notifies the operator.

- 3) operation parameter set phase

TS changes equipments configuration from test to operation.

- 4) antenna pointing phase

TS controls antenna equipment to orient satellite AOS direction.

- 5) satellite acquisition

After antenna orientation, TS waits for AOS of satellite. Antenna is driven according to the prediction. After the elevation angle of antenna arises above 2 degrees, TS checks the signal level. If the signal level is lower than the forecasted value, TS gives offset angle, which is equal to half of the beamwidth of antenna, to prediction angle for two axes and for plus and minus directions in order to avoid side beam lock. And TS searches the point where the highest signal level is obtained. On the condition of a good signal level, TS changes antenna drive mode from prediction to automatic tracking. After changing, TS confirms the mean and the standard deviation of tracking error angle.

If at first an acquisition antenna is used, TS changes to the main antenna after auto tracking become stable. Then TS confirms the difference of signal levels to avoid side beam

acquisition.

The received telemetry is demodulated, synchronized and sent to SOCS via the data transmission computer. By the way, the stored telemetry is kept in the base band equipment. This data is sent to SOCS later by stored telemetry transmission plan or remote operation from TACC.

6)uplink control phase

If command or ranging is planed in the network plan, TS controls the transmitter. The start timing of control is different between low orbit satellite case and geostationary satellite case.

In case of a low orbit satellite, TS starts to control the transmitter when the antenna rises above 5 degrees. In case of a geostational satellite, the period of uplink is indicated in the network plan. First, uplink frequency is swept in the direction which is determined by the predicted doppler. And TS verifies the satellite receiver status. After TS recognizes the satellite receiver lock, TS modulates the uplink.

If ranging or 1-way doppler is planed in the network plan, TS controls the ranging equipment. After the ranging data acquisition starts, TS confirms the measured value. If the difference between the measured value and the prediction value is large, TS stops data acquisition and retries once more.

7)operation terminate phase

In the case of a low orbit satellite, TS turns off the uplink when antenna elevation angle goes down below 5 degrees. And antenna drive mode is changed from auto tracking mode to prediction mode when antenna elevation angle goes down below 2 degrees. After the receiver loses satellite signal, TS tidies up equipments and terminates the network operation. In case of a

geostationary satellite, TS starts to tidy up equipments two minutes before network plan end time.

4.5 Network status monitor

TS always sends status of ground station equipment to TC. So, operators in Tsukuba can watch all station equipment status by the TC terminal. Also LC displays the line connection status, and that, operators in TSUKUBA can watch all NASDA network status.

4.6 Ranging data transmission

TS also sends the ranging data or the 1-way doppler data to NM via TC. They are kept in NM, and FDS gets them in batch operation.

4.7 Network control

As explained above, operators in Tsukuba need not directly control the network system. But, to cope with various situations, Network system allows operator to override network operations. Examples of control items are, interruption of network plan, extension of network plan, time offset of prediction, stored telemetry transmission, health check, and so on. Operators can control more detailed items as they were in front of ground station equipment.

5. THE SUPPORT AND INFORMATION SYSTEM AND ITS OPERATIONS

The support and information system is composed of systems as mentioned. The relation of the three systems is shown in Fig.3.

5.1 Flight dynamics system

Flight dynamics system(FDS) is responsible for determining the satellite orbit, making orbit maneuver plan, and providing satellite prediction. This system provides two kinds of predictions. One is the long

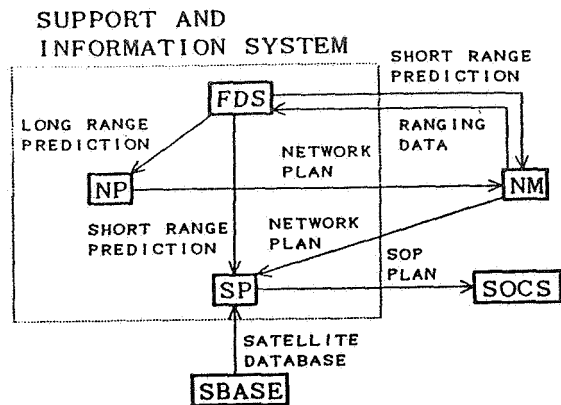


Fig.3 Data interface of the support and information system

range prediction and the other is the short range prediction. The former is used for network planning. The period of prediction is two or three months and the flight dynamics system makes it every month. The latter prediction is used for antenna driving and satellite operation planning. The period of prediction is 2 or 3 weeks. The interval of making prediction depends on the satellite. In case of JERS-1, the flight dynamics system provides it every day.

5.2 Network planning system

Network planning system(NP) is responsible for making network plan. NP accepts user's requests to use the network, assigns the stations and SOCS computers, and determines the network operation. NP makes monthly network plans one month prior to the start of the network plan. And NP renews network plans every week. Of course, if some emergency occurs, the network plan is renewed as soon as possible. NP has a function to print monthly, weekly, and daily network plans for network and satellite operators.

5.3 Satellite operation planning system

Satellite operation planning system(SP) is responsible for making

PLAN and SOP which is explained in section 3. Performance and operation of SP is different for each satellite operation planning system. In this paper, we explain the JERS-1 case.

SP of JERS-1 has the following functions.

- 1)read the event procedure written in Japanese.
- 2)pick up commands to send and telemetries to be confirmed in order to make SOP.
- 3)get network plan and determine operation style.
- 4)read operation request.
- 5)get prediction and make PLAN.
- 6)print revolution operation procedure.

The event procedure is a relatively short and definite procedure for satellite operation. It is prepared and SOP is made before launch.

In the operation phase, the operator of SP gets network plans from NM every week and determines the rough operation style. The operation style indicates whether the revolution is used for main operation or back up operation. Then, SP accepts mission operation requests and makes PLAN. But, parameter of time is not determined yet. SP determines the command time using the newest prediction every day. After making PLAN, SP sends it to SOCS.

6.CONCLUSIONS

NASDA has developed type-I SODS successfully. Presently, NASDA is preparing to operate the satellites which were launched before JERS-1 by type-I SODS. We are going to operate all satellites by type-I SODS next year. Meanwhile NASDA is developing the experiment equipment for inter-satellite communications. Next target of SODS is to integrate space network systems.

N94-23855

NASA/GODDARD SPACE FLIGHT CENTER'S TESTBED FOR CCSDS COMPATIBLE SYSTEMS

Richard D. Carper
Information Processing Division (560)
Goddard Space Flight Center
Greenbelt, Maryland 20771

ABSTRACT

A Testbed for flight and ground systems compatible with the Consultative Committee for Space Data Systems (CCSDS) Recommendations has been developed at NASA's Goddard Space Flight Center.

The subsystems of an end-to-end CCSDS based data system are being developed. All return link CCSDS telemetry services (except Internet) and both versions of the CCSDS frame formats are being implemented. In key areas of uncertainty multiple design approaches are being carried out. In addition, key flight-qualifiable hardware components, such as Reed-Solomon encoders, are being developed to complement the testbed element development.

The Testbed and its capabilities are described. The method of dissemination of the Testbed results are given, as are plans to make the Testbed capabilities available to outside users. Plans for the development of standardized conformance and compatibility tests are provided.

Key Words: Aerospace, telemetry, data systems, CCSDS, testbeds

1. INTRODUCTION

The National Aeronautics and Space Administration's (NASA's) Goddard Space Flight Center (GSFC) has instituted a significant testbed effort in the area of end-to-end data communications for space generated data. This program, the Advanced Orbiting Systems Testbed (AOST) Program, provides a bridge between development and widespread use of the Consultative Committee for Space Data Systems (CCSDS) Recommendations. Although named The Advanced Orbiting Systems Testbed, the Testbed in fact has the capability of creating and processing data compatible with both the Recommendation for Advanced Orbiting Systems (Reference 1) and of the Recommendation for Packet Telemetry (Reference

2). NASA has been supporting the development of CCSDS Recommendations since 1983. The Recommendations, being used by most future NASA missions, offer the promise of significantly reducing mission life cycle costs by standardizing data handling interfaces and functions in both space and ground systems, and of significantly increasing the reliability of operations due to repeated use of well tested and established procedures. In addition, standardization greatly enhances the ability of the various NASA space oriented subnetworks to operate with one another, and provides a solid basis for such interaction on an international basis where needed for international programs. Standardization allows the reuse of hardware and software, and makes it easier to support agency interoperation in national and international programs.

The AOST Program systematically addresses current impediments to using the Recommendations. The multi-faceted approach to the AOST Program will reduce the remaining costs and risks associated with the implementation of the Recommendations within NASA. With one exception, it is not a prototyping program. Its goal is the accumulation and dissemination of knowledge.

2. THE AOS TESTBED PROGRAM

The AOS Testbed Program has five main thrusts: developing and using the testbed itself, conducting a test program, performing directly related studies, developing critical flight-qualifiable components, and disseminating the knowledge gained in all the above.

2.1 Development of the Testbed

Development of the AOS Testbed draws upon the personnel and experience base of several different organizations with the GSFC. The flight-side equipment and the flight-qualifiable components are being developed by the Electrical Engineering Division. The front-end systems are being developed by the Data Systems Technology Division, as is one implementation of the Service Processors. Two other implementations of Service Processors are being developed by the Information Processing Division.

Services and Network Management is being developed by the MITRE Corporation. "Back-end" communications is being provided by the NASA Communications Division's independent NASA Open Systems Interconnection Protocols (NOSIP) Testbed. A Production Data Processing (PDP) capability will be provided to the Testbed as software by the PACOR II Project. The entire Test Program, including test execution monitoring and test results analysis is the responsibility of CTA Incorporated, which is also the program support contractor, providing documentation, coordination, and other management functions.

Development and use of the Testbed is divided into four phases or "Capabilities". Capability One provides the "unadorned" data handling functions for validation, architecture comparisons, and cost/performance evaluation. This capability is basically achieved. Capability Two sets these functions into an initial operations environment to identify performance degradation and to test interface definitions, requirements, and specifications. Capability Three completes the "rapid prototyping" into an operations environment and provides an initial set of standards conformance tests and systems compatibility tests. Capability Four completes the development of conformance and compatibility tests and provides the flight qualifiable components for flight project qualification.

2.2 The Test Program

The AOS Testbed test program is viewed as the primary means of obtaining the knowledge which is the objective of the Testbed. Testbed internal testing is performed to validate the achievement of each "Capability". Specific test periods (the majority of the Testbed development lifetime) are set aside for testing. The tests are formal and conform to a Master Test Plan and a Capability Test Plan; however, flexibility in schedule and sequence of testing is maintained in order to derive the greatest knowledge from the testing. Specific test program objectives are to:

- define and validate internal and external data and system management interfaces for AOS systems
- capture and disseminate knowledge gained in the implementation and testing of CCSDS Recommendations in an actual system
- raise issues associated with the implementation and testing of CCSDS Recommendations and communicate these issues to the CCSDS
- develop standard conformance tests and systems interface tests for use in acceptance testing by both flight and ground systems

complying with the CCSDS Recommendations

- establish cost/performance curves for various configurations

The AOS Testbed Test Program distinguishes five levels of testing complexity. These are:

- Element Tests - performed by implementors to ascertain a unit is ready for integration into testbed
- Compliance Tests - to determine that a unit does what it is supposed to do
- Interface Tests - to determine that two units operate properly together
- String Tests - to determine that three or more units operate properly together
- End to End Tests - to determine that the entire system operates properly

The Test Program has four types of tests. These tests may be conducted at any of the five test levels. The four types are:

- Functional Tests - to determine that the system/subsystem does what it is supposed to do
- Performance Tests - to determine that the system/subsystem performs at the speed or capacity or reliability, etc., that it is supposed to
- Regression Tests - to determine that the system/subsystem, after modification or upgrading, still performs properly
- Research Testing - includes investigations of choke points, influence of architectures on performance, cost-performance curve measurements, etc.

It is already clear that one of the most important products from the AOS Testbed will be the suite of tests that is being developed. The AOS Testbed is beginning a program for the development of standard test suites for "CCSDS Compatible" systems. The currently developed test suites will be restructured and enhanced to become standard test suites. It is planned these test suites will be made available to organizations that wish to do their own testing of existing or new systems. Both conformance and compatibility tests will be developed. Efforts are being made to accomplish this work in cooperation with the European Space Agency and possibly other international agencies. All results of this development effort will be provided to the CCSDS.

In addition to the AOS Testbed's internal test program, the Jet Propulsion Laboratory (JPL), the European Space Agency (ESA), the Canadian Space Agency (CSA), and the National Space Development Agency of Japan (NASDA) have expressed interest in

cooperative testing. A specific plan for testing with ESA is in its final preparation stage.

2.3 Related Studies

By far the most significant study effort being supported by the AOS Testbed is its participation in the ad hoc inter-Center Space Operations Service Infrastructure (SOSI) working group. Most of the members of this group are also members of CCSDS Panel 3. The SOSI group is developing a top-down, end-to-end graphical model that can be used to:

- Develop service concepts
- Develop operations concept
- Identify cross support points, protocols, and formats
- Discuss cross support issues and interfaces
- Be a basis for CCSDS Panel 3 work

The CCSDS Packet Telemetry, Telecommand, and AOS protocols are tuned to operate across the space link, terminating in a ground station. The CCSDS space link protocols conserve bandwidth (and allow high speed data processing) by using managed information instead of in-band signaling and by minimizing redundancy. The Ground System must extend the services provided by these space link protocols to user application processes. Usually the user applications are in a Ground-based End System (GES) that is provided data from several distant Ground-based Spacecraft Access (GSA) systems over commercial or private data networks. The GSA is an intermediary between the space link and various end systems on the ground. GSAs typically provide services to many missions. The scheduling, management, and reporting necessary to allow multiple missions to receive services from multiple GSAs -- efficiently and correctly -- is a key aspect which the SOSI model is intended to illustrate. A key function of the GSA is to apply annotation information before the data unit leaves the GSA. The annotation information is of two types:

- SDU Information which is specific to each sublayer of the service:
 - information not known prior to ground receipt (e.g., loss of sync; Reed-Solomon Corrections; Sequence Discontinuities, ground receipt time)
 - information carried in one sublayer but needed in another sublayer (e.g. Spacecraft ID)
 - known prior to ground receipt but lost if the service were to be extended beyond the RF terminal .
- Managed information: distributed systems will be easier and cheaper to operate if information which is managed over the

space link subnet due to bandwidth limitations, is signaled over the ground system.

In addition, an agency implementing a GSA may wish to attach agency specific annotation information for purposes of accounting and fault isolation (e.g., system IDs).

The SOSI model and related concepts are being developed in cooperation with the AOS Testbed. Both architectural issues and data format development are worked between the "top-down" approach of the SOSI and the "bottom-up" experimentation within the Testbed. A more detailed discussion of the SOSI work may be found in Reference 3.

A key element in this cooperative work is the content and format of the annotation information. The annotation information is applied to each data unit to be delivered by the GSA. The annotation mechanism as so far developed by the AOS Testbed, is called a Space Operations Service Data Unit (SOSDU) header (when the header is added to the data unit, the combination is called a SOSDU). The SOSDU header currently contains a generalized label, a SOSDU label which is the same for all SOSDUs, a set of Service specific parameters which are different for each of the Services, and possibly optional agency and/or network unique fields. The SOSDU header is a relatively early stage of development and is expected to change significantly over the next several months. A major revision to the format is expected to be completed within the next month.

2.4 Flight-Qualifiable Components

The objective of this part of the AOS Testbed Program is to generate key products to promote flight project use of CCSDS Recommendations. These products were planned to include flight qualifiable coding components, flight qualifiable frame level components, and flight qualifiable packet generators/buffers. A good foundry run has already been achieved of a single chip, high speed Reed-Solomon Encoder. The chips will be made available through the University of New Mexico. Unfortunately, the current budget situation has required a halt in the funding of this activity. It is hoped that some FY93 funds can be found to continue the work at a lower level of activity, or that it can be re-started in FY94.

2.5 Dissemination of Knowledge Gained

The primary vehicle for the dissemination of the knowledge gained through the various activities of

the AOS Testbed is the annual "Workshop for CCSDS Implementors". The first of these was held on November fourth and fifth of this year at the Goddard Space Flight Center, and was very successful. Over 300 persons attended, representing 44 companies, 2 universities, 4 NASA Centers, NASA Headquarters, and the Department of Defense. The attendance was far in excess of our expectations, demonstrating a very broad interest in CCSDS implementations. (A reason for this wide-spread interest can be seen in Table 1, Spacecraft/CCSDS Compatibility.) The attendance was so large that we found it impractical to actually have a "workshop", and it was in fact a seminar. We will change the format for the next workshop, at least part of the time breaking up into interest groups, thereby providing a true workshop environment.

3. AOS TESTBED ARCHITECTURE

There are two major architectural drivers considered within the AOS Testbed Program. First is the distribution of functions due to network and facility drivers. Second is the allocation of functions to subsystems within a major "infrastructure" facility. The first is being addressed by the Space Operations Service Infrastructure (SOSI) group (see above). So far, the two drivers seem to lead to a highly compatible allocation of functions. The testbed architecture was selected to expose potential cross support (inter-network) points, provide easy access to inputs and outputs of major functional blocks, and to allow easy and standardized (serial) interconnection among the elements. The testbed elements, shown in Figure 1, represent all of the subsystems of the return side of an end-to-end data system (forward link capability is not presently included or funded). The rate performance target for the ground side is 20 mbps peak, and for the flight side, 80 mbps peak. Simulators are used for the end users systems, both onboard and on the ground.

A more detailed drawing of the Flight Elements of the Testbed is shown in Figure 2. The digital video equipment string is not yet completed, but is expected to be added in late spring of 1993. A detailed drawing of the ground side configuration is given in Figure 3. It is in this Figure that the major architectural features are discernible. The Local Area Networks (LANs) of the Testbed may, in real-world implementations, be Wide Area Networks (WANs). The same is true of the second Testbed LAN. The multiple implementations of the Services Processors allow for experimentation in both the geographical distribution of Services Processors and in the maximization of resources utilization in a heterogeneous multi-user ground facility.

Results from testing and study in the AOS Testbed may be found in Reference 4.

4. ACKNOWLEDGMENTS

The author wishes to express his sincere thanks to the members of the AOS Testbed implementation team. Their dedication and hard work is producing invaluable information which will influence the design of space data systems for years to come. And while doing this, they have also made this a most enjoyable and personally rewarding project.

Special thanks are due to Mr. Charles Fuechsel, of NASA Headquarters, whose original idea and consistent support has made the AOS Testbed a reality.

And last, my gratitude to Ms. Penny Newsome, head of the Program Support Contract, whose unflagging efforts and exceptional management abilities have made a major contribution to the success of this program.

5. REFERENCES

1. CCSDS, 1989. *Recommendation for Space Data System Standards. Advanced Orbiting Systems, Networks and Data Links: Architectural Specification*. CCSDS 701.0-B-1, CCSDS Secretariat, NASA, Washington, D.C.
2. CCSDS, 1987. *Recommendation for Space Data System Standards. Packet Telemetry*. CCSDS 102.0-B-2, CCSDS Secretariat, NASA, Washington, D.C.
3. Pietras, John and Theis, Gerhard, 1992. *A Reference Model for Space Data System Interconnection Services*. In Space Ops 1992 Symposium Proceedings
4. Newsome, Penny A. and Otranto, John F., 1992. *The Advanced Orbiting Systems Testbed Program: Results and Lessons Learned to Date*. In Space Ops 1992 Symposium Proceedings

TABLE 1 SPACECRAFT/CCSDS COMPATIBILITY

SPACECRAFT	LAUNCH	FRAME	R-S CODING	PACKETS	PKT TIME	CODE FWD LNK
EUVE	92	NO	NO R-S	YES		
SAMPLEX	92	YES	CCSDS CONV	YES	YES	YES
FAST	94	YES	CCSDS CONV	YES	YES	YES
SWAS	95	YES	S	YES	YES	YES
SOHO	95	YES	YES	YES	YES	YES
XTE	96	YES	YES	YES	YES	YES
TRMM	97	YES	YES	YES	YES	YES
ACE	97	YES	YES	YES	S	S
SMEX 4	96	YES	S	YES	YES	YES
SMEX 5	97	YES	S	YES	YES	YES
SMEX 6	98	YES	S	YES	YES	YES
EOS-AM	98	YES	YES	YES	YES	YES
EOS-PM	99	YES	YES	YES	YES	YES
SSFP	96	YES	YES	YES	S	YES
MARS OBSRVR	92	YES	YES	YES	YES	NO
CASSINI	97	YES	YES	YES	YES	YES

S = UNDER STUDY

FIGURE 1 AOS TESTBED ELEMENTS

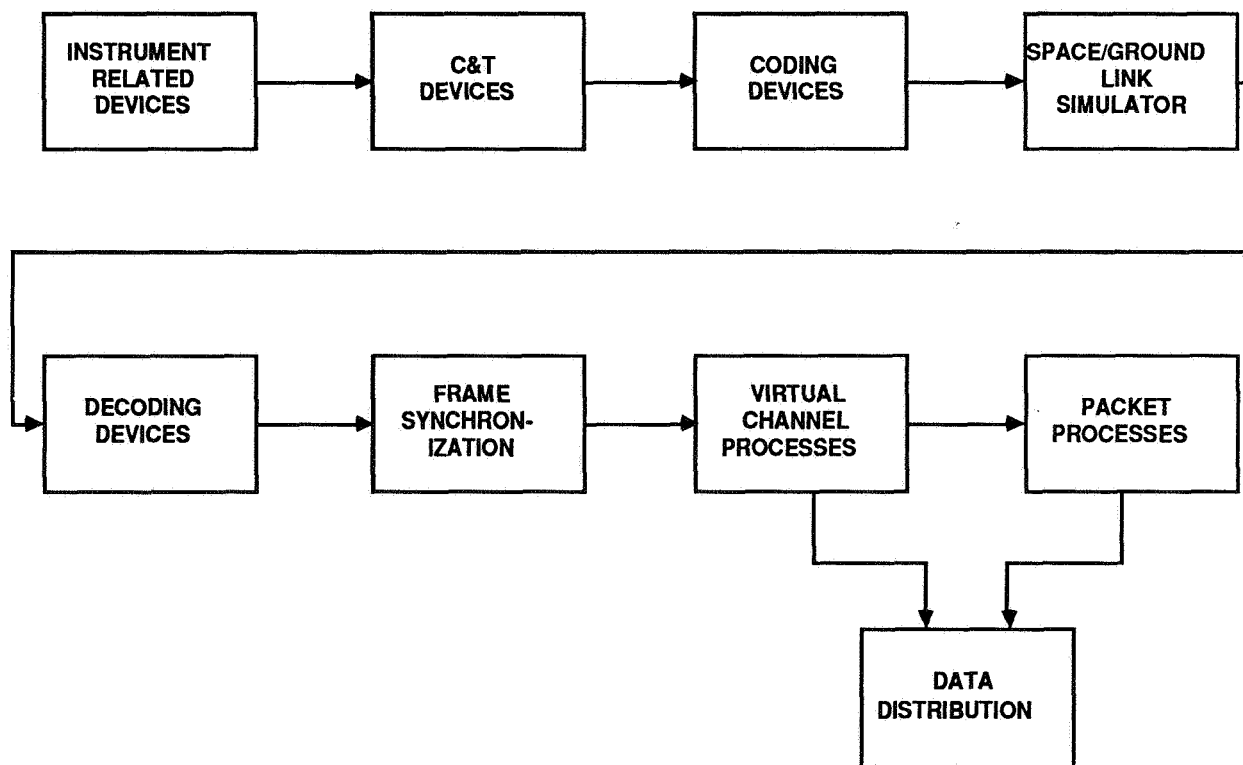


FIGURE 2 AOS TESTBED FLIGHT SIDE CONFIGURATION

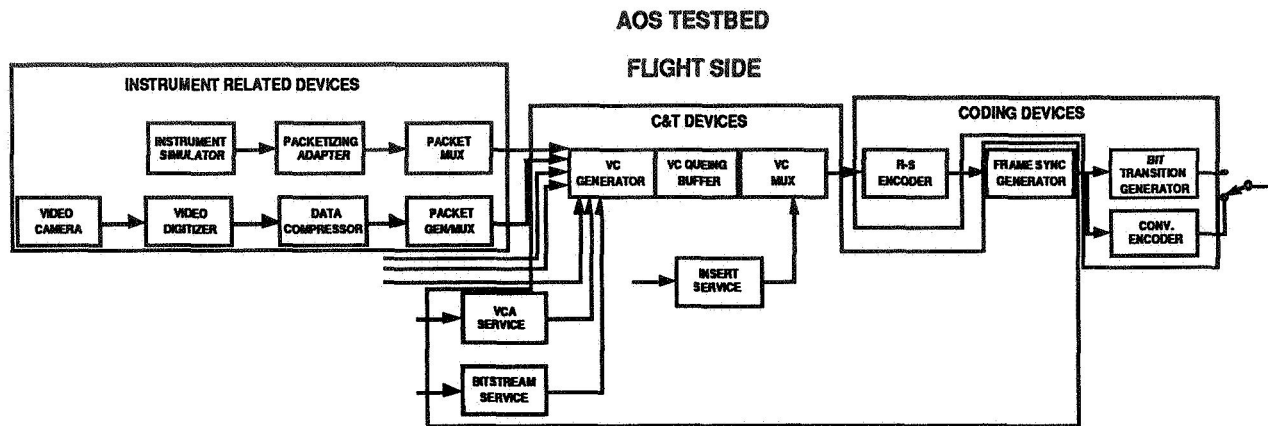
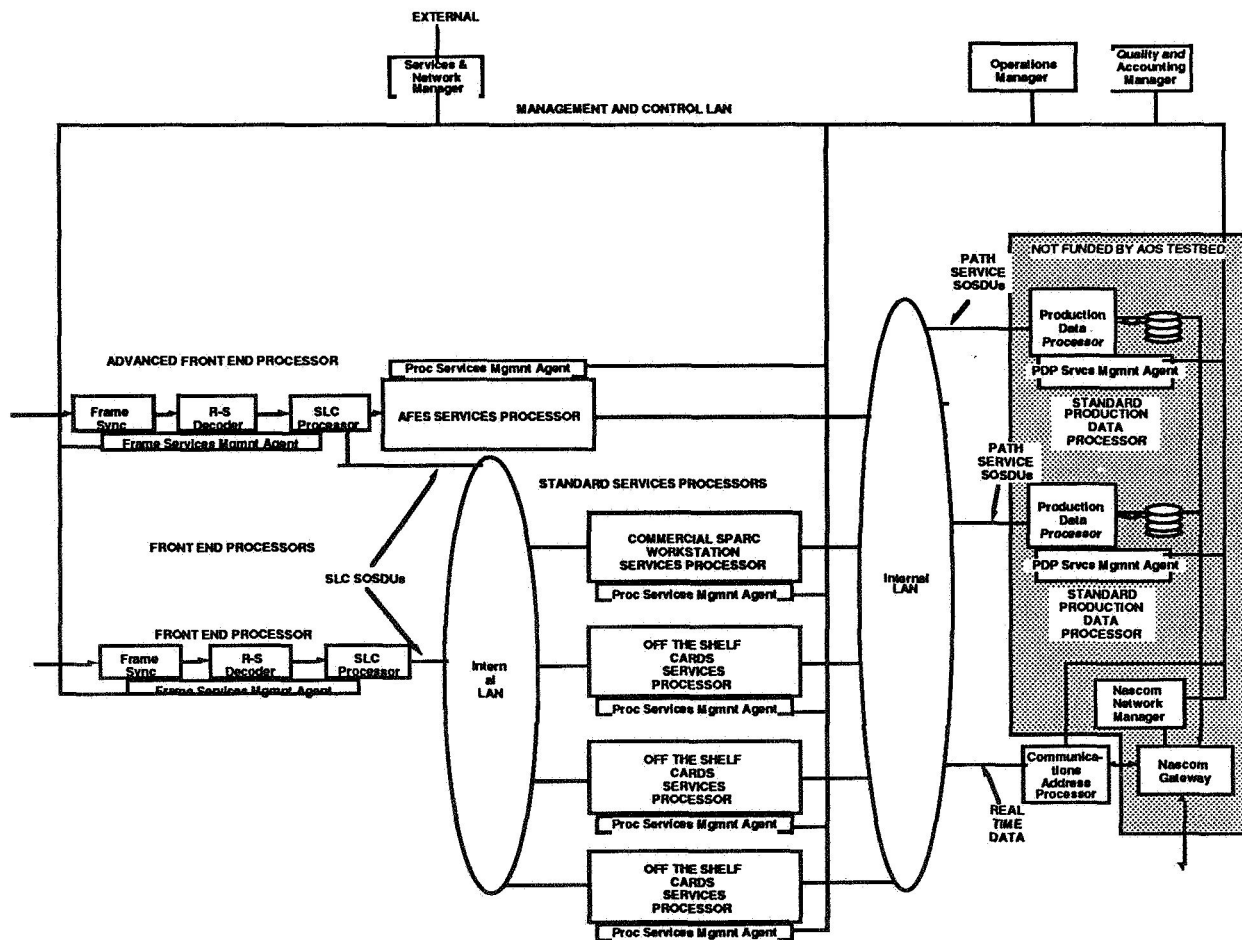


FIGURE 3 AOS TESTBED GROUND SIDE CONFIGURATION



SPOT4 OPERATIONAL CONTROL CENTER (CMP)

G. Zaouche

N 94 - 23856

Centre National d'Etudes Spatiales (CNES), 31055 Toulouse CEDEX, FRANCE

ABSTRACT

CNES(F) is responsible for the development of a new generation of Operational Control Center (CMP) which will operate the new heliosynchronous remote sensing satellite (SPOT4).

This Operational Control Center takes large benefit from the experience of the first generation of Control Center and from the recent advances in computer technology and standards.

The CMP is designed for operating two satellites at the same time with a reduced pool of controllers.

The architecture of this CMP is simple, robust and flexible, since it is based on powerful distributed workstations interconnected through an Ethernet LAN.

The application software uses modern and formal software engineering methods, in order to improve quality and reliability, and facilitate maintenance.

This software is table driven so it can be easily adapted to other operational needs.

Operation tasks are automated to the maximum extent, so that it could be possible to operate the CMP automatically with very limited human interference for supervision and decision making.

This paper provides an overview of the SPOT4 mission and associated ground segment. It also details the CMP, its functions and its Software and Hardware architecture.

Key Words: Control Center, workstations, standards, flexible, automation.

1. INTRODUCTION

The SPOT system, conceived and designed by the CNES, the French Space Agency, is an earth observation system. Its aim is to manage and provide images acquired by SPOT satellites and transmitted to ground stations.

The SPOT satellites family consists of :

- SPOT1 launched in February 1986;
- SPOT2 launched in January 1990;
- SPOT3 launch planned in Sept. 1993;

The continuity of this service is now ensured by the CNES SPOT4/5 programs up to 2000 and later (SPOT4 launch is planned for 1995).

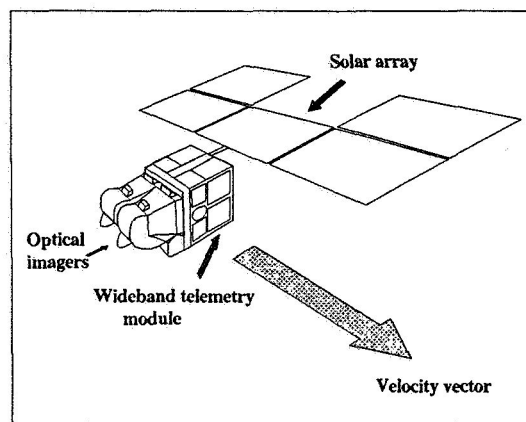
Each component of the SPOT4 system has been designed to take into account the satellite evolutions (new sensors, ...) and the improvements provided by the acquired experience (6 years) and the increasing knowledge of the SPOT IMAGE clients' wishes.

2. SPOT4 SYSTEM DESCRIPTION

2.1 SPOT4 Satellite

The SPOT4 spacecraft is composed of :

- . a standard platform that can be used for remote sensing missions other than SPOT;
- . a payload consisting of optical imagers and an image telemetry system



SPOT4 belongs to a new generation of satellites compatible with the previous one.

The main characteristics of SPOT4 satellite are :

- identical to SPOT1,2,3 :

- . sun synchronous, polar orbit;
- . mean altitude 832 km, cycle period : 26 days;
- . nodal period : 101.4 mn, number of tracks : 369;
- . maximum pass time : 785 s;
- . width in vertical viewing : 60 km (117 km twinned instruments), adjustable viewing axis : $\pm 27^\circ$;
- . high resolution images : 10 and 20 m.

- main evolutions :

- . life duration in orbit : 5 years;
- . mass increased : 1/3, passengers as PASTEL, ...
- . supplementary spectral band (Medium Infra Red),
- . registration of all the spectral bands;
- . capacity of Telemetry x 2 and Telecommand x 3;
- . 50 Mbits/s data storage during 40 mn;
- . new tape recorders, ...

2.2 SPOT4 Ground Segment

The evolutions of the ground segment take large benefit from the experience of the first ground segment designed for SPOT1. The new implementation tries to split the different independent functions and to affect them to each center.

To reduce the exploitation costs, a larger automation is requested for each component.

The Operational Control Center (CMP) is one of the elements of the SPOT4 ground segment.

The Ground segment is divided in two large parts corresponding to the two distinct missions it must carry out :

- monitoring and managing the satellite;
- exploiting the received image data.

The first mission is ensured by the following components :

- the CNES (S-Band) Stations (belonging to the CNES 2GHz network) follow the satellite during the visibility period and interface it for all the data exchanged with the Control Center;
- the Operational Control Center (CMP) monitors the satellite and its passengers in real time

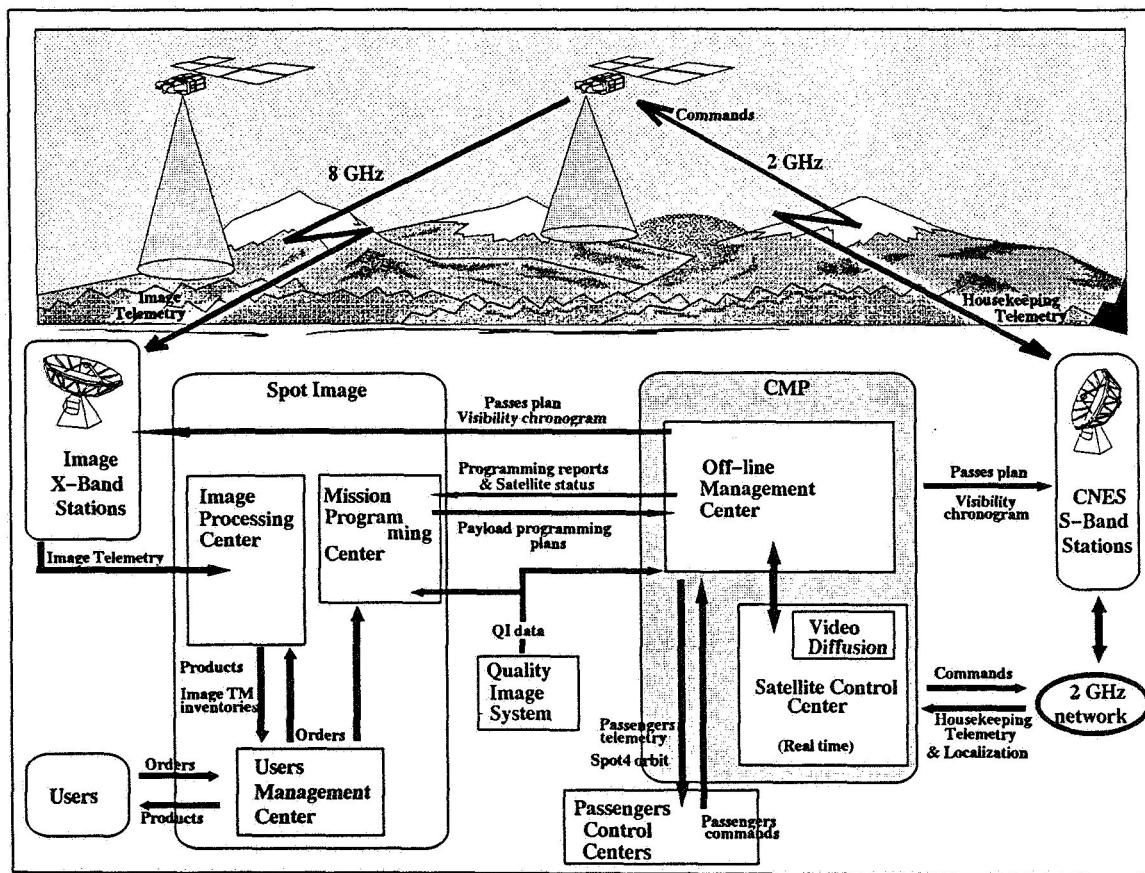
by receiving the house-keeping telemetry, the localization data and transmitting telecommands to the satellite. It operates the planning of activities for the platform, the passengers and the payload, on the base of programming requests;

- the Quality Image System (QIS) checks, through repetitive measurements and calculations, the image performances in rough mode or after pre-processing.

The other components exploiting satellite image data are :

- the Image Reception (X-Band) Stations record the image telemetry on high density tapes;
- the Users Management Center (CGC) is in charge of relations with clients (reception of orders, product distribution, ...);
- the Mission Programming Center (CPR) programs the payload to optimize the load of the satellite and the use of the viewing instruments;
- the Image Processing Center (CAP) systematically archives the received image telemetry and pre-processes the required products.

Data flows between the main components of the SPOT4 ground segment are summarized on the following diagram.



3. SPOT4 OPERATIONAL CONTROL CENTER

3.1 Mission Requirements

The SPOT4 Operational Control Center (CMP) is located in Toulouse Space Center, France. It is presently designed to provide simultaneous control over 2 satellites.

The overall objectives are :

- technological monitoring of the satellite and control in real time and off-line modes through management of a technological data base;
- satellite commanding for on board configuration management;
- orbit calculation, in orbit positioning and station keeping maneuvers;
- payload programming, either in routine mode (generation of commands corresponding to the plan programmed by the CPR), or in special mode (under CMP responsibility) in case of payload failures, ...
- short and long term planning of all ground operational activities (availability of receiving stations, etc...);
- generation of all orbit data necessary for external facilities (Image Reception Stations, passengers control centers, ...);
- video images broadcasting to operation control rooms for the launch time and specific orbital maneuvers.

3.2 Operation Requirements

The CMP has to be operational in the following exploitation phases (for 2 satellites) :

- system qualification,
- launch and acquisition phase,
- flight acceptance,
- nominal exploitation (5 years foreseen).

The CMP may be operated automatically, only attended by a single operator during the routine phase in nominal exploitation. The exploitation activities run in cadence with the passes and must be planned and prepared in advance in order to run without operator's intervention.

The CMP is designed to follow 20 passes a day (two satellites). It must process the housekeeping telemetry TM (more than 2500 parameters by 16s frame) without loss in real time. The CMP has also strong requirements for the orbit maintenance to avoid degrading the optical performances of the system.

3.3 Functional Architecture

The functional architecture of the CMP is designed to provide both real time and off-line functions. It is based on a time driven concept.

The CMP is divided in two subsystems :

- the Satellite Control Center (CCS),
- the Off-line Management Center (CGS).

3.3.1 Satellite Control Center (CCS)

It is in charge of the real time (or near real time) functions and all its activities are linked to the passes. Its main functions are :

. before the pass :

- automatic test of the station,
- automatic preparation of the passes, taking into account commands from various subsystems of the CGS, scheduling the commands according to the visibility plan and generating a command plan to be automatically sent to the satellite.

. during the pass :

- reception of housekeeping telemetry,
- monitoring in real time and display of the processed parameters,
- acquisition of localization measurements for orbit monitoring,
- broadcast of TM synoptics to the video distribution system (launch and critical phases),
- automatic transmission of the command plan (previously prepared) to the satellite.

. after the pass :

- transmission to the CGS of the received and validated TM, localization measurements, acknowledged commands,
- telemetry replay of a previous visibility at original speed, at full speed or step by step under operator's control.

3.3.2 Off-line Management Center (CGS)

The Off-line Management Center performs off-line functions which are distributed in several subsystems :

. Satellite Technological Monitoring (STS)

- archiving telemetry and updating the technological data base;
- providing the CCS with TM for replay;

. Orbital Maneuvers Management (OMGS)

- orbit computation and control, delivery of orbit data to operations;
- automatic generation of routine commands for orbit management and maneuvers programming.

. Dump and TC Block Handling (DUTCH)

- on board software management, dumps analysis and on board configuration management.
- trending analyses

. Payload Management (CGCU)

- payload instrument resources management and programming.

. Passengers Management (GTPA)

- passengers technological monitoring;

4. HARDWARE ARCHITECTURE

4.1 Requirements

The CMP hardware architecture is designed to answer the following requirements :

- perform the mission,
- obtain a high level of reliability and availability,
- evolve easily,
- manage breakdowns by using a flexible design.

4.2 Choice

The architecture is based on powerful distributed workstations, interconnected through an Ethernet bus. Each subsystem is implemented on separated workstations. The level of redundancy is the subsystem level.

- workstations : HP 9000 series 835 and HP 9000 series 720 and 750 computers, RISC architecture;

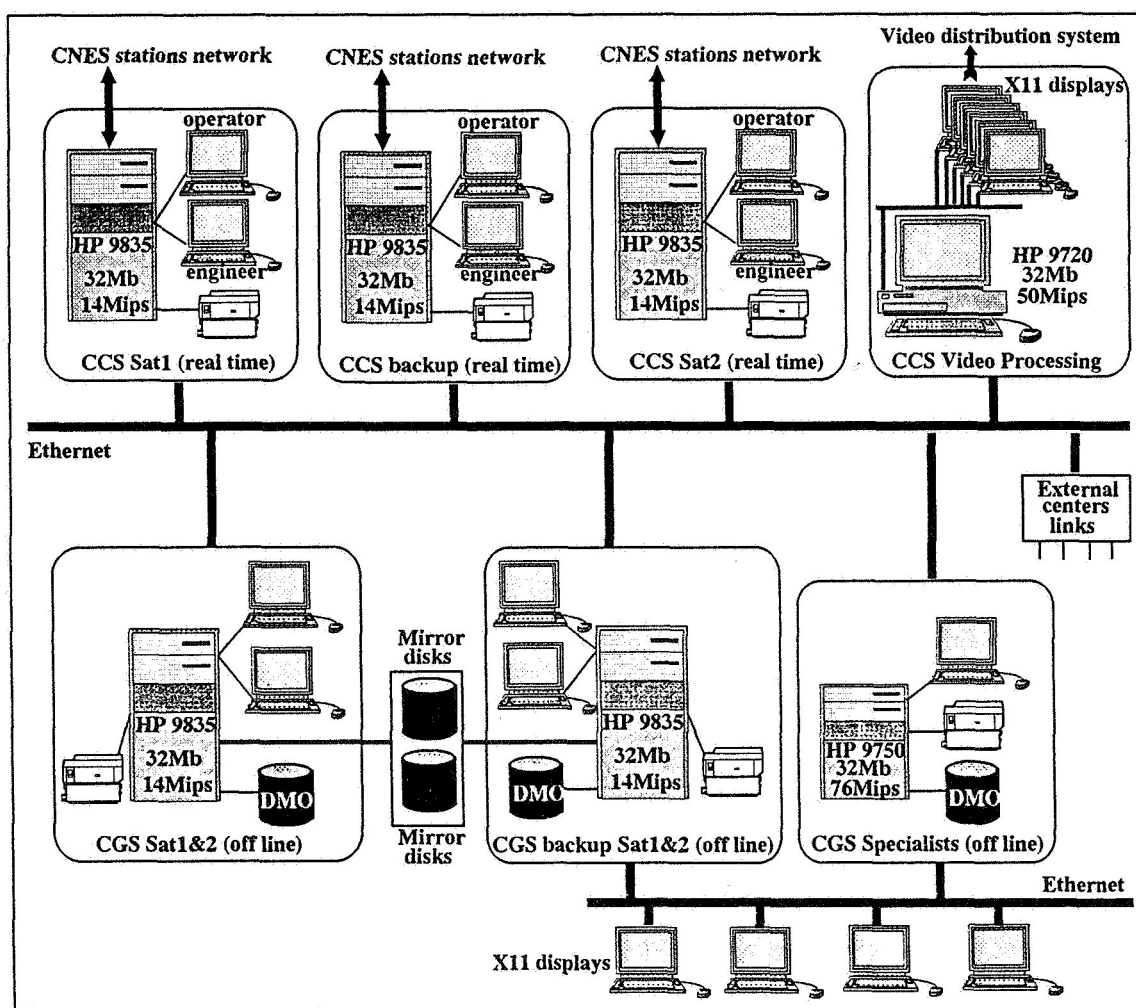
- operating system : HP/UX SYSTEM V compatible;

The use of modern technology and standards has been maximized :

- 32 Mbytes memory, optical magnetic disk;
- disk mirroring management, bitmap, X11 displays;
- graphical interface : X11, OSF/MOTIF;
- coding source languages : C / FORTRAN;
- internal communication on TCP/IP, NFS;
- X25, HDLC drivers, ...

4.3 Evolutions

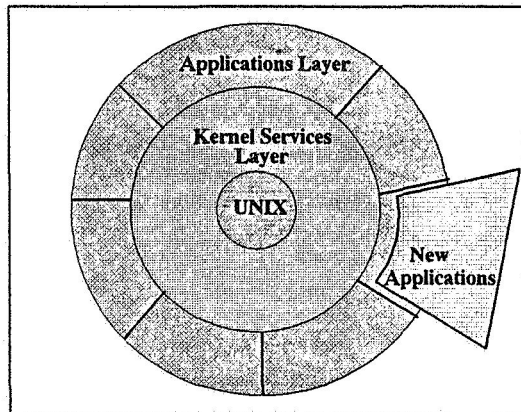
Evolutions taking into account increases in the mission requirements (providing control over a third satellite or increasing the number of TM synoptics to be broadcasted to the operation control rooms) are easy to design with such a hardware architecture. It consists often in adding a supplementary workstation with a potential resizing of the Ethernet network.



5. SOFTWARE ARCHITECTURE

The software architecture is composed of two layers above the System layer (HP-UX, OSF-MOTIF) :

- the Kernel and Services layer (CNS);
- the Application layer.



5.1 Kernel and Services Layer (CNS)

This layer consists of :

- libraries for system calls standardisation and common services access :
 - . message management (broadcasting to files, windows, printers,...),
 - . file management (ASCII files can be text edited),
 - . inter-process communication (message queues, shared memory, semaphores),
 - . inter-computer communication,
 - . time management (shifted time).
- utilities (general purpose applications)
 - log file exploitation,
 - . hardware and software configuration control,
 - . archive tape management,
 - . computers time synchronisation.
- Agenda and Procedure management
 - . easy and ergonomic work schedule definition,
 - . automatic applications execution as defined in the work schedule,
 - . standardised start/stop and control of application programs over the network,
 - . recovery steps on failure occurrence.

Pilote	Editor	Procedure	Journal de Bord	Session
CCS1				
CCS2				
STS1				
OMGS				
		Station AUS		
Debut fenetre 07:12:00	Durée fenetre	2h	PGT n°1	Repere : 00:00:00 Passage
				Cacher Initialiser

An Agenda (Ref.1) work plan scheme

5.2 Application layer

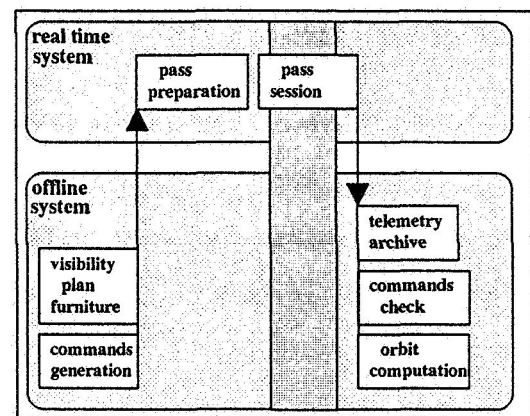
Each operational function is ensured by a software package called "procedure". These procedures are overlaid above the Kernel and Services layer.

A procedure is an autonomous task designed to run without operator's intervention and to be executed on one of the CMP's workstations. It can be activated automatically by the Agenda (Ref.1) or manually by the operator through an ergonomic man machine interface.

Building a work plan for one or several days consists in defining on the Agenda (Ref.1) work schedule the wished procedures.

Skeletons of work plan are available for the routine exploitation mode.

Example of a daily routine procedures sequence :



5.3 Standardisation and Reliability

5.3.1 Man Machine Interface

The Man Machine Interface is implemented on X11/Motif standards. High resolution screens (19" bitmaps and X11 displays) are available for the implementation of all the animated synoptic windows and the alphanumeric windows.

Ergonomic specifications (based on the OSF/MO-TIF style guide) were drawn up to ensure that the CMP screens had an uniform design. In most cases mock-ups were built to be analysed by the future Center users.

5.3.2 Adaptation Capability

For each developed procedure, very strong rules have been established to increase the software adaptation capability :

- independence from satellite type by designing the software according to a table driven concept (characteristics in an Oracle database);

- systematic use of ASCII "SR6-10" files for the external or internal interfaces to facilitate integration and maintenance.

- independence in the real-time part (CCS) between a producer of data and the consuming task(s) (producer/consumer protocol);

- high level of software quality by using up-to-date and formal software engineering methods :

- * SADT for the functional phase -

tool : ASA,

- * "Abstract Machines" for the design phase - tool : CALLIOPE,

- * coding sources rules, test coverage and quality metrics - tool : LOGISCOPE,

- * non-regression at process level and full validation at system level,

- * configuration management since the integration phases - tool : CMF.

6. CMP EXPLOITATION

The new generation CMP has taken into account, at the beginning of the development, the cumulated experience (from SPOT1 and SPOT2) of the exploitation team.

6.1 Human factors

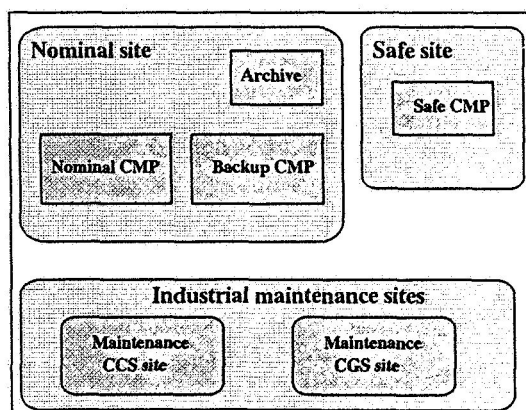
Only one operator will be present in routine operations (1 operator/team; 2 teams/day). The operator interventions take place on failures or non conformances.

The engineering exploitation team will consist of configuration, documentation, quality assurance managers, ground and on board specialists.

About 25 persons will work at the CMP, including the management team.

Along with the operators team, satellite specialists work in the CMP for exceptional checks or in flight acceptance phase.

6.2 Configurations on Site



6.3 Documentation

As a result of the development phase, a fully documented software is provided to the exploitation team.

The documentation is produced through an automatic generation (interface manuals, design manuals, ...).

By using automatic generation of operational plans, hard copies of Agenda work plans, procedures libraries, etc ..., the exploitation documentation will be easier to manage than the previous documentation.

6.4 Failures

If a failure occurs, all the system is designed to avoid an untimely reaction from the operator.

The Agenda stops the procedures and all the means of investigation are given to specialists (graphical displays, printer outputs, log file, ...).

Each default occurring in communicating with the satellite (on board reception default, incorrect transmission conditions) or inside the CMP needs a manual intervention of an operator or a specialist.

In automatic mode, at any time the operator can break the automatic working of the CMP in order to command and control manually the required procedures.

Recovery procedures are pre-defined and easily started manually or through the Agenda.

7. STATUS

The CMP development should be completed by mid 1993.

Some of its subsystems (as the CCS) are already accepted.

8. REFERENCES

1. Fratter, I. 1992. Agenda: a task organizer and scheduler. In SpaceOps 92.

SESSION C

MISSION-SPECIFIC SYSTEMS

CO-CHAIRPERSONS

K. W. Ledbetter, NASA, USA

W. G. Meeks, JPL, California Institute of Technology, USA

Summary	157
W. G. Meeks	

C.1 CNES Organization for Station Positioning of Geostationary Satellites	159
J. Dulac, CNES Toulouse Space Centre, France	

C.2 Mission Activities Planning for a HERMES Mission by Means of AI-Technology	165
U. Pape, G. Hajen, N. Schielow, and P. Mitschdörfer, Deutsch Aerospace, ERNO Raumfahrttechnik GmbH, Bremen, Germany; F. Allard, ESA European Space Research and Technology Centre, Noordwijk, The Netherlands	

C.3 A Quick Uplink, Expandable Executable for the NSSC-I of the Hubble Space Telescope	173
G. T. Foley and E. A. Citrin, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA	

C.4 Ground Data System for Space Flyer Unit (SFU)	179
T. Yamada, S. Sasaki, and K. Ninomiya, ISAS, Sagamihara, Japan	

C.5 Radioastron Flight Operations	185
V. I. Altunin, Astro Space Center, Lebedev Physical Institute, Russian Academy of Sciences, Moscow, Russia; K. G. Sukhanov, Babakin Center, Lavochkin Association, Moscow, Russia; K. R. Altunin, JPL, California Institute of Technology, Pasadena, California, USA	

C.6 The Ulysses Spacecraft Control and Monitoring: Concepts and Realities	191
P. Hamer, Cray Systems, Bristol, UK; N. Angold, ESA European Space Operations Centre, Darmstadt, Germany	

C.7 Ulysses Log 1992	197
R. García-Pérez, ESA European Space Operations Centre, Darmstadt, Germany	

C.8 Galileo Spacecraft Anomaly and Safing Recovery	203
R. R. Basilio and D. M. Durham, JPL, California Institute of Technology, Pasadena, California, USA	

C.9 Magellan Spacecraft and Memory State Tracking: Lessons Learned, Future Thoughts	209
A. W. Bucher, Martin Marietta Astronautics Group, Denver, Colorado, USA	

SUMMARY — SESSION C

W. G. MEEKS
Jet Propulsion Laboratory

All the scheduled papers were presented except "The TRMM Prototype for Science Data Information," by B. E. Lowrey and G. K. Kim of NASA Goddard Space Flight Center. The authors were unable to attend because of travel restrictions.

The session was well attended; audiences ranged between 40 and 65 people for the nine papers presented. The highest responses were for C.6, "The Ulysses Spacecraft Control and Monitoring: Concepts and Realities," and C.8, "Galileo Spacecraft Anomaly and Safing Recovery." The Ulysses presentation described making the transition from Earth orbit operations — which had been a part of ESOC's experience — to planetary operations during the flight out to Jupiter and back to the vicinity of the Sun's poles. The Galileo

presentation described how the recovery from "safe mode" was reduced from 13 days at the outset to the present 40 hours by refining procedures.

The paper by V. I. Altunin and K. G. Sukhanov of Russia and K. R. Altunin of JPL entitled "Radioastron Flight Operations" (C.5) was interesting and well received by audience members, who were reflective of the end of the Cold War.

Compliments must go to the organizers of the symposium. These sessions should continue to provide this kind of forum for mission operations, which I consider a separate discipline.

CNES ORGANIZATION FOR STATION POSITIONING OF GEOSTATIONARY SATELLITES

C.1

Jean DULAC

N 9 4 - 2 3 8 5 7

CENTRE NATIONAL D'ETUDES SPATIALES
TOULOUSE, FRANCE

ABSTRACT

Since 1975, the Toulouse Space Centre (a technical establishment of the French Space Agency, CNES) has successfully brought 15 geostationary satellites on to station.

During these 17 years of experience, an organization of human and material resources has been built up that ensures a very high level of reliability in the execution of these station positioning operations.

The main characteristics of this organization are a rigorous definition of the roles and responsibilities of each person involved, very detailed operations documentation, and methodical preparation of the operations.

Key Words : Geostationary satellite, station positioning, operations documentation, organization

1. INTRODUCTION

It was in December 1974 that CNES and the DFVLR successfully brought SYMPHONIE-A, the first European geostationary satellite, on to station. This first station acquisition was followed by that of SYMPHONIE-B in August 1975.

This early experience proved very useful when CNES had to make the preparations for bringing the first French telecommunications satellite, TELECOM 1, on to station at the beginning of the '80s.

The technical and human organization set up at that time showed its efficiency when TELECOM 1-A was correctly positioned on station in August 1984.

The principles of this same organization have since been used by CNES to bring 13 successive satellites on to station :

- TELECOM 1B
- TDF 1
- TELE-X
- TELECOM 1C
- TDF 2
- INMARSAT 2 F1
- INMARSAT 2 F2
- TELECOM 2A/INMARSAT2 F3(*)
- ARABSAT 1C
- TELECOM 2B/INMARSAT2 F4(*)
- HISPASAT 1A

(*) brought on to station simultaneously

Although the contractual and technical contexts of these station acquisition operations were very different, the organizational principles stayed the same, demonstrating their capability to adapt and develop.

In this article, we present the three most characteristic aspects of this organization :

- rigorous definition of the responsibilities of those involved in the operations,
- very detailed operations documentation,
- methodical training for operations.

PRECEDING PAGE BLANK NOT FILMED

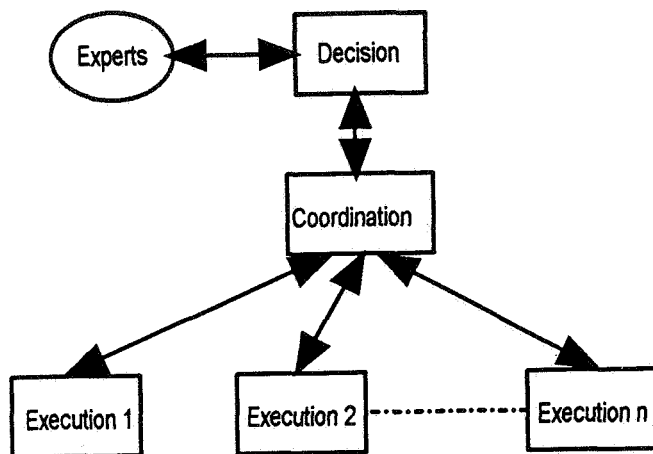
2. OPERATIONAL ORGANIZATION

2.1 Those concerned

There are three levels in the organization :

- Decision
- Coordination
- Execution

as shown diagrammatically below.



At the decision level there is a team of experts (generally from the industrial organization building the satellite) responsible for helping the decision-makers.

2.2 Modes of operation

There are two possible modes of operation while a satellite is being brought on to station:

- *nominal mode* : the satellite behaves as expected; the redundancy in the ground equipment is sufficient to ensure replacements in cases of failure.
- *degraded mode* : the satellite's behaviour is abnormal; the level of failures in ground equipment is such that execution of the operations cannot be ensured with sufficient safety.

In *nominal operating mode* : the operations are performed in accordance with the pre-

established plans and procedures under the leadership of the coordination entity, the decision-making entity only intervening to authorize the starts of the most critical phases of the mission (launch, apogee manoeuvres, etc.).

In *degraded operating mode* after the anomaly has been identified, the experts draw up an emergency strategy and the corresponding executable procedures with the heads of the entity concerned.

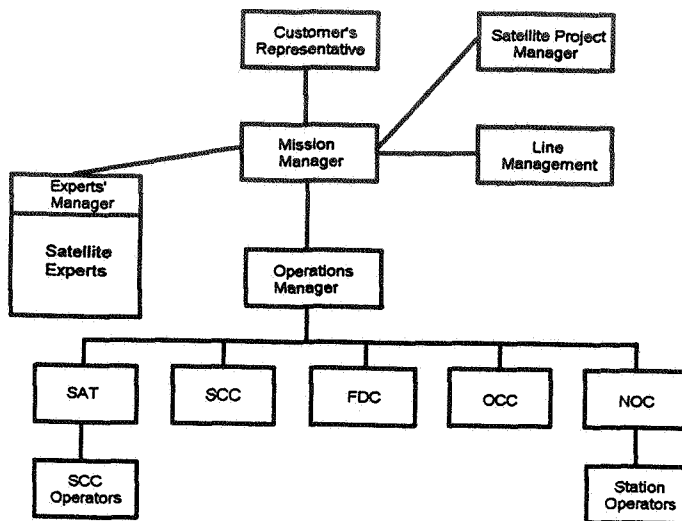
After authorization by the decision-making entity, the procedures are executed by the entities concerned under the orders of the coordination entity.

2.3 Organization for TELECOM 2 station acquisition

As an example, the figure below gives the organization set up to bring the TELECOM 2 satellites on to station.

The functions of each person concerned are as follows :

- *The Mission Manager* is responsible for how operations progress and keeps the Customer's Representative informed. He sees to the application of degraded case procedures.
- *The Customer's Representative, the Satellite Project Manager and the line management* participate in decision making for degraded situations.
- *The Operations Manager* conducts the station acquisition operations and coordinates the activity of the operational entities
- *The Satellite Engineer (SAT)* is responsible for the execution of actions on the satellite when the situation is nominal.
- *The Head of the SCC* is responsible for the running of the Satellite Control Centre.
- *The FDC Manager* is responsible for the elaboration of orbit and attitude manoeuvres; he supplies the Operations Manager with the data needed for their execution.



The OCC Manager is responsible for the acquisition of localization measurements from the various networks, for orbit restitution and for station designation.

- *The NOC Manager* coordinates the activity of the stations in the network used for bringing the satellite on to station.

- *The Experts* monitor the satellite's behaviour and suggest corrective action if behaviour is abnormal. *The Experts' Manager* coordinates their activity.

The people concerned work in different rooms which may be far apart; communications are provided by a system of interphones and video image transmission.

3. OPERATIONS DOCUMENTATION

3.1 At decision level

The Decision Plan describes the decision-making process during operations, particularly for cases where operations do not progress as expected.

3.2 At coordination level

The General Operations Plan describes the overall operations organization and the ground system configurations. It serves as a

reference for drawing up the Specialized Operations Plans (SOPs).

The Control Plan contains the linked chronological sequence of all the procedures needed to successfully bring the satellite on to station. The Operations Manager is in charge of its unfolding throughout the course of operations.

The Flight Plan is a graphic overview of the Control Plan. For each orbit it shows :

- the stations in view
- the manoeuvres to be executed.

3.3 At execution level

The Manoeuvre Operations Procedures (MOP's) precisely define the operations to be carried out on the satellite : telecommands to be sent, telemetry monitoring. The MOP's are executed under the responsibility of the Satellite Engineer.

The Specialized Operations Plans (SOP's) individually define the organization, configuration and, possibly, the timeline specific to each entity.

The Ground Procedures state the implementation procedures for the various equipment or software used for each entity.

3.4 Documentation for Operational Qualification

The Operational Qualification Plan defines the qualification principles for the ground system, the specifications to be verified and the list of exercises to be carried out.

Each exercise is defined in an *Operations Order* and the results of the exercise are recorded in an *Exercise Report*.

4. PREPARATION FOR OPERATIONS

4.1 Technical assembly

Technical assembly consists of linking together the main components of the system and testing that they work properly. This phase starts after acceptance of the components developed specifically for the mission.

The main components to be taken into consideration are, for the case of the TELECOM 1 ground system :

- The Specialized Control Centre (SCC)
- The Flight Dynamics Centre (FDC)
- The 2 GHz Network (CNES and NASA stations)
- The Orbit Computation Centre.

A document called the Technical Assembly Plan defines, for each test :

- the objectives
- the initial conditions
- those involved
- the procedures to be applied
- the planning.

This phase is very important; it checks that the ground system is working properly and allows the operations teams to familiarize themselves with its use.

4.2 Operational Qualification

The purpose of *Operational Qualification* is to demonstrate that the ground system for station acquisition is capable of performing the station acquisition operations.

It has three principle objectives :

- to train all staff,
- to validate the operations documentation,
- to demonstrate the system's capacity to carry out the operations in accordance with the Flight Plan.

The programme is made up of three levels, each level being composed of several exercises having a common aim.

Level 1 :

Training exercises in the implementation of the operational configuration of the ground system with training in :

- diagnosing ground failures
- changing ground configuration

Level 2 :

Exercises in the progression of individual operations procedures for station acquisition manoeuvres, the entities concerned and satellite experts involved in the test being brought into play. The purpose of these exercises is to give the teams overall training in the time sequence, qualify the documentation and demonstrate the ground system's ability to carry out the operations under operational conditions. All the exercises take place in nominal spacecraft and ground operating conditions, although a few degraded cases will also be requested.

Level 3 :

Exercises in the progression of linked sequences of station acquisition under real conditions with the experts present and gradual introduction of degraded cases for the ground and spacecraft, so as to test the teams' capacities to deal with anomalies while respecting the operations procedures.

Each exercise is described in an operations order, which states :

- the test identification number,
- the date and time of the start of the test,
- the aim of the test,
- the duration of the test,
- the configuration requested,
- the simulation facilities required,
- the initialization case to be used by the simulator,
- the entities concerned,
- the documentation used,
- the information specific to the test,
- the staff concerned.

At the start of each test, a short briefing by the Operations Manager confirms the information contained in the operations order and the test conditions.

At the end of each exercise, a debriefing by the Mission Manager brings together the Operations and Entity Managers and the Customer's Representative for an immediate report on how the exercise went. This also allows the Quality Manager to take the anomalies into account.

The decision to accept or rerun the exercise is taken at the debriefings.

Operational Qualification generally lasts two months for the first flight model of a family of satellites and a month or less for the following ones.

During Operational Qualification, great care must be paid to managing the changes made to the operations documentation.

4.3 Waiting in Operational Condition

This phase corresponds to the period of a few weeks (nominally 4 to 2) between the end of Operational Qualification and launch. In principle, the configuration is frozen and only those changes that are absolutely indispensable are made to the system.

A few exercises are performed to enhance team training.

CONCLUSION

The organization set up at CNES for bringing geostationary satellites on to station is the result of long experience. It has proved its capacity to adapt to varied technical contexts (Networks, Satellites, Launcher) and different contractual relationships.

MISSION ACTIVITIES PLANNING FOR A HERMES MISSION BY MEANS OF AI-TECHNOLOGY

N 94 - 23858

U. Pape¹, G. Hajen¹, N. Schielow¹, P. Mitschdörfer¹, F. Allard²

¹ Deutsche Aerospace - ERNO Raumfahrttechnik GmbH
P.O. Box 105909, 2800 Bremen, Germany

² European Space Agency - ESTEC
P.O. Box 229, 2200 AG Noordwijk, The Netherlands

ABSTRACT

Mission Activities Planning is a complex task to be performed by mission control centers. AI technology can offer attractive solutions to the planning problem. This paper presents the use of a new AI-based Mission Planning System for crew activity planning which has been set up in the course of an ESA project in cooperation with an industrial consortium led by ERNO Raumfahrttechnik. Based on a HERMES servicing mission to the COLUMBUS Man Tended Free Flyer (MTFF) with complex time and resource constraints approx. 2000 activities with 50 different resources have been generated, processed and planned with parametric variation of operationally sensitive parameters. The architecture as well as the performance of the mission planning system is discussed. An outlook to future planning scenarios, the requirements and how a system like MARS can fulfill those requirements is given.

Key Words: crew activity planning, time- and resources constraints, distributed planning, rule based planning.

1. MISSION SCENARIO & OVERVIEW

The case study conducted is based on a HERMES servicing mission to the COLUMBUS Man Tended Free Flyer [ref 1]. The mission scenario includes a HERMES launch rate of 2 per year. The involved ground segment comprises a Hermes Flight Control Center (HFCC), an MTFF Control Center (MSCC), a coordinating Combined Mission Control Center (CMCC) as well as User and Engineering Support Centers. Crew activity planning for such type of missions has to consider the following main aspects:

- o the HERMES spaceplane system:
 - launched by an ARIANE 5 launcher
 - Mass of payload 3.0 metric tons

- nominal mission duration: 10 days (Free Flyer Servicing: 6 days)
- 3 crewmembers (commander, pilot, mission engineer)

- o the Man Tended Free Flying Laboratory:
 - 2 segment Pressurized Module (PM) for accommodation of 16 P/L racks (12 m³)
 - Resource Module (RM) docked to PM
 - Propulsion subsystem for altitude and attitude control
 - ARIANE 5 launch vehicle
- o MTFF servicing concept:
 - Corrective/Preventive Maintenance
 - Replenishment of consumables
 - Inspection and cleaning
 - Implementation of growth features
 - P/L Upgrading, repair, reconfiguration & replacement of payload facilities
 - P/L Resupply and collection of samples & consumables
 - P/L Checkout and set-up of P/L processes
 - Servicing by IVA, EVA, Remote Manipulation and combined EVA and Remote Manipulation activities.

- o Mission Cargo Sets includes all serviceable items for a single servicing mission, including:
 - permanent travellers (to be flown each mission)
 - life limited items (to be exchanged at end of life)
 - stochastic items (corrective maintenance)
 - consumables (e.g. gases & fluids)
 - airborne support equipment (ASE) & tools

2. THE PLANNING APPROACH

Crew tasks like internal servicing of subsystems or payloads, external servicing of subsystems, crew sustaining operations, etc. can be broken down into subtasks and activities. The resulting

hierarchical structure of crew activities has to be converted into a planning database which can then be accessed by the planning system.

The nodes of the planning database contain all the relevant planning information, like: earliest and latest start time, permissible time window, used resources (power, crew member, sun / eclipse phases, communication channels, etc.), antecedent links to other activities etc.

Resources are specified by : name, type, availability and max capacity.

The idea is first to build a generic planning database which can serve as a library of generic tasks, subtasks and activities to be copied for reuse and customization for a specific mission, thus to reduce significantly the effort for the compilation of planning data. The contents of the specific database depends totally on the Mission Cargo Sets.

Consistency checks of the specific DB (e.g. missing or inconsistent antecedent links) are supported by the planning system itself.

Such a system can be easily used as a valuable tool for quick assessment of design changes by planning with parametric variation of operational sensitive parameters.

3. THE PLANNING SYSTEM

3.1 The MARS Scheduling System

The scheduling of the activities selected for a specific mission was performed by means of the AI based mission planning tool MARS (Mission Activities and Resource Scheduler [ref 2]) developed by ERNO Raumfahrttechnik GmbH. The design of MARS explicitly takes into account the complexity aspects of current and future space missions. It has been steadily evolved in tight cooperation with operational users. Within the ESTEC contract NEPTUNE (New Expert System for Users in a Network Environment [ref 3]) MARS was enhanced by more general resource handling capabilities, a rule system, configuration management features and distributed planning capabilities. It now includes all features necessary to successfully apply it not only to space missions planning but to any kind of large project scheduling.

The main modules of MARS comprise the Database Editor and the Scheduler each of which will be briefly described in the following.

The Database Editor is a graphical interactive tool to allow the efficient creation and modification of activity and resource descriptions, rules to direct the scheduling process and constraints. Activities are hierarchically structured as ex-

plained in chapter 2 and handled by the Activity Editor [Fig. 1].

Activities are described by the following major attributes:

- o Full activity name and abbreviation, position in the activity hierarchy.
- o Earliest and latest possible start time and due date (if fixed by problem at hand).
- o Minimum and maximum duration.
- o Any number of antecedent activities and min/-max delay times to them (logical dependencies).
- o Priority, keywords and comments by the user.
- o Interdependencies to other activities (mutually exclusive set of activities).
- o Any number of resource requests in the form of: resource-name and resource request profiles.

The Resource Editor allows the specification of resources as well as the generation of resource availability profiles. Any number of different resources can be introduced by the user. Each resource can be of one of the following types:

- o **Sharable** resources represent system states. E.g. the execution of several activities can be restricted to happen during certain day or night periods (SUN-ECLIPSE Phases).
- o **Depletable** resources can be used to describe incremental consumption of a given capacity, e.g. Battery or Storage Place.
- o **Reusable** resources can only be used by one user at a time but are available afterwards, e.g. Tools and Energy. Conventional project planning tools only provide this single type of resource.
- o **Tri-state** resources allow to specify mutually exclusive sets of activities (e.g. activities of one set cannot run in parallel with activities of another set). It is a combination of sharable and reusable.

Optionally resource requests can be defined that are used permanently even after termination of the given activity. Resources can also be generated by activities. Additionally a calendar defining the general working hours and off days can be defined as in any conventional project planning system.

For the generation of schedules and timelines adhering to all given constraints a set of rules (heuristics) can be specified using the Rule Editor. Those rules influence or direct the scheduling process. E.g. it is possible to schedule in

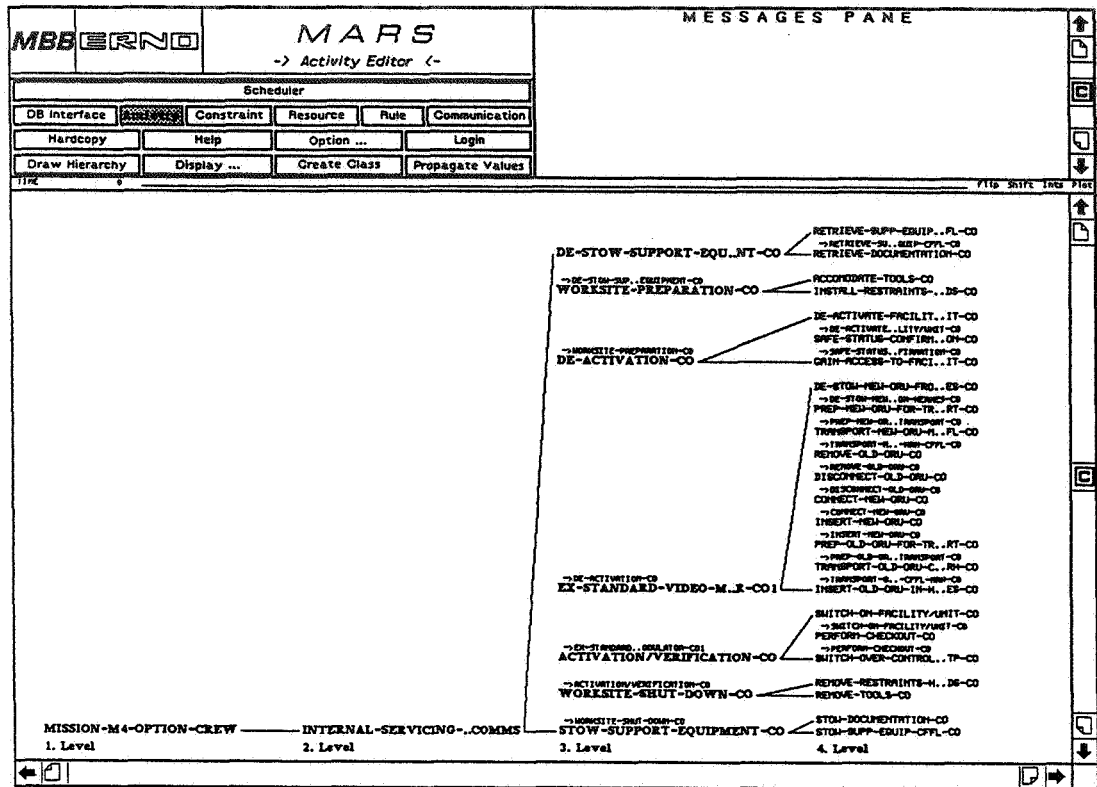


Fig. 1 The Activity Editor

priority a selected set of activities or to schedule in priority the most resource consuming activities. Using this feature of MARS the user is very flexible in defining his/her own preferences or goals to be achieved by the scheduling process.

Four different rule types exist:

- o **Scheduling Rules**, which select the set of activities for the next pass of the scheduling process.
- o **Activity Selection Rules**, which influence the selection of the next to be scheduled activities.
- o **Rescheduling Rules**, which influence the re-scheduling process and
- o **Backtracking Rules**, which direct the scheduling process during backtracking situations.

Scheduler

The basic problem in any scheduling process involving (limited) resources is defined by the requirement to satisfy both temporal and resource constraints simultaneously.

For MARS therefore a new method has been developed, called **Time Constraints Propagation Mechanism (TCPM)**, that transforms all symbolic time constraints on and between activities into

a computationally much more efficient algebraic representation covering all time relations defined between activities. At first "time constraint windows" are computed for each activity with respect to the totality of all temporal constraints imposed on the plan. Scheduling of activities only within these time windows guarantees that no temporal constraint conflicts will occur at any point in the scheduling process. Thus futile attempts to schedule activities outside these windows are avoided, thereby speeding up the scheduling process considerably and increasing the chance of generating a conflict-free schedule. It is the task of the TCPM to compute and dynamically update these permissible time windows after each scheduling step.

This separation between handling time and resource constraints leads to a scheduling strategy whereby at each point in time, starting from left to right on the time axis, only those activities which lie within their permissible time windows and whose resource and target requests are satisfied by the resource and target availability profiles for the total duration of the activity are identified. Obviously each of these activities individually represents a candidate for scheduling. However, usually not all candidates can be scheduled simultaneously, but only candidate

- o Any type and duration of resource violations and involved activities are graphically displayed.
- o The user can graphically shift an activity forward or backward in time and view the induced conflicts (or conflict-free areas for scheduling) dynamically displayed by the "co-moving" scheduler (on forward shift) or Backtrack Module (on backward shift).

A typical example of a schedule generated by MARS is depicted in Fig. 2.

To demonstrate the ability of MARS to interact with different types of already existing or future mission specific data (e.g. configuration and planning data) and to support the concept of a generic planning database the mission planning data for this project has been stored in an external database whose structure reflects the foreseen COLUMBUS Mission Database Application (MDA). For this purpose the Mission Database Application Software (MIDAS) was developed based on the commercial ORACLE DBMS to provide MARS with the required data. The information maintained by MIDAS consists of:

- o Display of the schedule and composite timelines in the form barcharts and Gantt diagrams.
- o Resource Usage, giving the percentage resource exploitation over the total mission time (as an indication of the schedule quality).



- o all scheduling relevant data, such as activity and resource definitions, resource availabilities, generic library of tasks and subtasks and Master Timelines (MTLs) as the result of the scheduling process,
- o and support data, such as the description of the Mission Cargo Sets (MCS) and ORACLE relevant data (database dictionary).

MIDAS allows the interactive data entry and maintenance. It supports the hierarchical name tree concept which provides for expressing the relation of objects in a tree like structure. Planning data for a specific mission can be generated by instantiating the generic activity descriptions depending on the selected mission cargo set. A version concept was implemented to allow tracking of data changes over time and provide access to actual or older versions of database item descriptions. Database access control features provide limited data manipulation privileges for certain data to certain user groups. Finally report generation, database recovery and consistency check functions have been implemented.

3.3 The Toolset Architecture

Since the MARS system and the MIDAS application were intended to be used in a distributed environment running on different physical machines (for this project MARS ran on a Symbolics Ivory LISP coprocessor board residing in a SUN 4/330 with the Genera 8.0 operating system whereas MIDAS was running on a SUN 3 with SunOS 4.0.3) interface S/W was developed to allow communication between MARS and MIDAS in a client - server architecture.

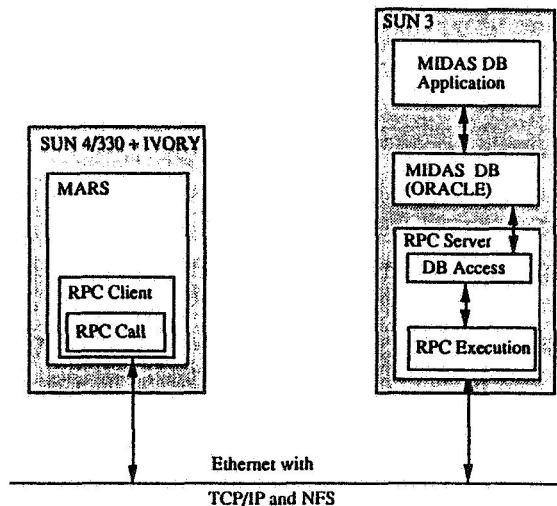


Fig. 3 Toolset Architecture

Both machines were connected by Ethernet utilizing the Network File System (NFS) S/W from

SUN which also supports Remote Procedure Calls (RPCs). Using this mechanism data can be exchanged between MARS (as the client) and MIDAS (as a database server). For example if MARS wants to read data from the database it specifies as input parameters for a READ RPC the type (key) of data and those requested data are transferred as return parameters. To perform the execution of the RPC's on the SUN a server software was implemented. This server handles the execution of all the specified procedures. It accesses the database via SQL statements. The complete architecture is depicted in Fig. 3. The server and all remote procedures are implemented in PRO*C or directly in C, while the calling of procedures from within MARS is implemented in LISP.

4. PLANNING DATA ANALYSIS

The approach to generate planning data includes 3 main steps:

- o Definition of generic crew activities library which can be reused for any mission cargo set.
- o Definition/selection of the mission cargo set to be transported by HERMES and operated by the crew including all items for:
 - a selected nominal MTFF servicing mission
 - a selected MTFF contingency mission
- o Definition of specific crew activities database based on generic data and the selected mission cargo set.

Only those crew activities were subject for planning which start with docking of HERMES to MTFF and ending with de-docking of HERMES.

4.1 Generic Operations Definitions

All generic crew activities have been structured hierarchically into phase, task, subtask and activity level. The 1st level of the resulting crew activities breakdown is shown in Fig. 4.

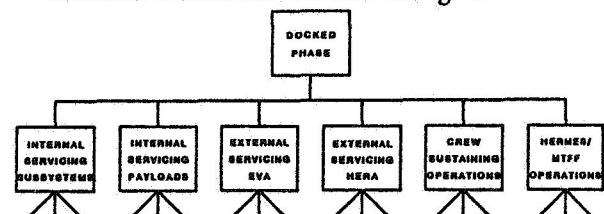


Fig. 4 Activity Breakdown (1st Level)

The complete breakdown resulted in a total of 80 subtasks and 431 generic activities. An example of the activities breakdown established for

the exchange of a failed video modulator (internal servicing of communication subsystem) is also given in Fig. 1.

4.2 Mission Cargo Sets

The required items/tasks to be installed/accomplished by the crew are determined by the HERMES and MTFF system capabilities as well as by the servicing concept, strategy and constraints (see chap.1). The 20 Mission Cargo Sets (MCS) for the first 10 years of operation have been generated by means of a Monte Carlo Simulation. Out of these a representative MCS has been selected for the detailed planning. For the 114 ORU's of the selected mission a rough accommodation analysis to derive the operational properties of the different ORU's has been performed including an Exchange Task Time Assessment as well as an ORU criticality analysis to derive priority information to be used by the planning system in case of conflicts.

4.3 Specific Operations Definition

Based on the generated MCS information a mission specific database has been set up. The major steps were:

- o definition of a suitable task structure to allow the re-use of the existing substructures of the generic database to the extent possible,
- o setup of the lower level task/activities structures by:
 - importing
 - modification
 - duplicationas appropriate for the specific mission demands,
- o tuning of the generic planning information imported from the generic database according to the mission specific constraints, e.g. time constraint information, resources and resource usage,
- o specification of the resource availabilities,
- o definition of the targets (orbital opportunities) in the form of time-histories.

The resulting mission specific database includes 32 Tasks, 399 Subtasks and 1741 Activities.

The same exercise was performed for a contingency mission which has been based on the nominal mission defined before. The assumptions for this mission have been:

- o repair of a micro meteorite impact in the outer shell of the MTFF

- o repair to be performed by means of EVA
- o exact location and extent of the problem is not known a priori, thus requiring inspection activities with remote manipulator
- o specific repair tools had to be included

The database for this mission includes 24 Tasks, 252 Subtasks and 966 Activities.

5. CREW WORKLOAD ANALYSIS

The generation of conflict free schedules of activities defined in the mission database has been performed for the nominal as well as for the contingency mission.

Another important analysis was the variation of operationally sensitive parameters in order to investigate selected contingency scenarios and options to change mission rules.

For the nominal mission the following variations of operationally sensitive parameters have been performed:

o Crew Task Change

The Mission Engineer has been removed from all subsystem servicing activities and the Pilot and Commander got the same priority to perform subsystem servicing.

o MCS Reduction

Some cargo items could not be accommodated physically in HERMES. The corresponding crew activities have been removed from the schedule.

o Crew Sickness

The Mission Engineer is not available on the first servicing day for about 4 hours due to space sickness.

o Communication Dropout

RF - Communication between HERMES / MTFF and ground is interrupted for 3 hours on the 4th mission day.

The generated schedule for the nominal mission showed that with the a priori defined crew allocation to the mission activities only 35 % of the science payloads could be processed. Valuable statistical functions for exploitation of the available crew resources resulted in the following numbers:

- o Commander = 61.6 %
- o Pilot = 62.9 %
- o Mission Engineer = 84.9 %

These numbers gave indications to assign more tasks in the science payload area to the Commander and to the Pilot, which resulted in

Planning Results Summary	Nominal Mission "M4"	M4 Planning Options					Contingency Mission "M4C"
		Crew Task Change	MCS Reduction	Crew Sick-ness Case A	Crew Sick-ness Case B	Comms Dropout	
Total Mission Time [d/h/m] [min]	11 d 8h 27 m 1 6 3 4 7	8d 1 1 5 2 0	10d11h48m 1 5 1 0 8	11d7h53m 1 6 3 1 3	11d12h30m 1 6 5 9 0	11d11h11m 1 6 5 1 1	8d 1 1 5 2 0
△ Time to " M4" [min]	N / A	- 4 8 2 7	- 1 2 3 9	- 3 4	+ 2 4 3	+ 1 6 4	N / A
Database Size: Phases Tasks Subtasks Activities	1 32 399 1741	1 32 399 1741	1 32 385 1676	1 32 399 1741	1 32 399 1741	1 32 399 1741	1 24 252 966
Crew Usage: Commander Pilot Mission-E.	61,6% 62,9% 84,9%	89,4% 87,2% 84,6%	62,9% 63,9% 85,5%	62,1% 63,5% 83,6%	61,3% 62,7% 84,2%	61,0% 62,4% 84,1%	74,4% 75,5% 81,4%
Subsystem Servicing: Start Time End Time Serviced ORUs	1d2h35m 5d8h16m 100%	1d2h35m 5d11h16m 100%	1d2h35m 5d5h50m 100%	1d2h35m 5d6h12m 100%	1d2h35m 5d11h50m 100%	1d2h35m 5d10h40m 100%	3d3h25m 7d2h14m 100%
Payload Servicing: Start Time End Time Serviced ORUs	5d8h16m 11d2h4m 35%	1d2h55m 6d10h18m 100%	5d5h50m 10d5h25m 48%	5d6h12m 10d14h33m 35%	5d11h15m 11d6h5m 30%	5d10h40m 11d4h48m 32%	N / A

Tab.1 Summary of Planning Results

a 100 % accomplishment of all tasks. Table 1 indicates the results obtained by the other variations.

6. OTHER SYSTEM APPLICATIONS

MARS has shown its versatility and operational capabilities in a number of different application areas. The first important application field was the planning of manned and unmanned space missions on tactical as well as on executional level. This also covers the application as described in this paper. Furthermore, MARS was used to perform dynamic payload accommodation based on information contained in payload configuration databases (such as the COLUMBUS P/L Database CPDB) and to analyse alternative accommodations [ref 4].

In a study on future autonomous space systems MARS was used as a building block for autonomous scheduling and rescheduling. Complemented by a diagnosis system and spacecraft simulator the system was able to demonstrate the capabilities of a fully autonomous spacecraft that reacts to malfunctions by diagnosing its internal status, initiates appropriate recovery actions and finally replans the rest of its mission according to the new situation [ref 5].

Current activities comprise the development of interface software and the rework of planning

data originally compiled for the MIPS (Mission Planning System) used to plan the next German Spacelab Mission (D2) to be launched early next year. For this application MARS is intended to be validated during the mission in terms of its performance and operational capabilities by comparing MARS with the built-in scheduler (ESP, Experiment Scheduling Program) of the MIPS system in replanning situations. For this purpose MARS is extended to be able to deal with the same input and output data formats as the ESP system. Former studies have already shown the advantages of MARS as an automatic and interactive scheduling tool wrt ESP, but this will have to be justified during that real mission.

Another recent project is the utilisation of MARS for the planning of the ARIANE4 second stage and booster production and integration that is performed at ERNO premises.

Under the product name PARS (for Project Activities and Resource Scheduler) the MARS system will also be made available commercially on SUN workstations until end of this year.

Distributed Planning

The COLUMBUS mission planning concept is currently still under definition. MARS can support an operational scenario which is based on two important principles:

- o the distributed Ground Segment Infrastructure and
- o the hierarchical system for its management

According to the hierarchy of the distributed planning environment there will exist a tree structure of logical nodes each with its own MARS system, each responsible for the planning of a specific logical node or mission center in the hierarchy. Within a logical node the activities are structured itself in a tree hierarchy as described for the mission planning database editor. In a top-down procedure each center allocates resource envelopes to its lower level centers over the network, so that these can generate their own local schedules and timelines. Afterwards these local timelines will be integrated in a bottom up approach into the higher level timelines. This MARS features can also be applied to any type of project planning, where the problem at hand must be decomposed and solved by several planning teams, working closely together.

7. CONCLUSION

The results of the project have clearly shown that such a planning system can be an essential tool throughout the lifecycle of a space project, starting with the initial definition of the space infrastructure until and within the operational phase. In the very early project phases trade-offs can be performed to assist in the design of the spacecrafts and payloads. What-if scenarios can be worked out to identify bottlenecks in the system design by analysing operational scenarios (timelines).

The planning system used in this study compared to other commercially available systems, e.g. for project planning, has the ability to handle different kinds of resources, e.g. to handle stocking tasks by filling intermediate stores. It can be interfaced with other software and databases through its programming language interface and the possibility exists to extend its intrinsic functionality to implement any upcoming and very specialised requirements, e.g. for additional project specific resource types.

These features would allow the cost effective application of MARS in other very demanding scheduling domains where resources are a limiting factor and the timespan for execution of tasks is restricted, such as:

- o Production planning, scheduling of production tasks, interleaving of multiple production lines

- o Transportation planning
- o Logistics planning

Future alternative hardware platforms for the planning tool MARS will be any UNIX based system that supports a Common Lisp environment with multitasking capability and X window system interface using CLIM and CLOS (e.g. SUN workstations). This will achieve the goal of principal hardware independence for MARS and derivatives (PARS) and will be finished in 1992.

8. ACKNOWLEDGEMENTS

The following organisations have contributed to the implementation and application of the described system:

SENER, Bilbao - Spain
CISER, Rome - Italy

9. REFERENCES

1. Pape, U.; Allard, F.; et al. 1991. Final Report HTL HERMES Mission Analysis and Timeline Engineering. Noordwijk, the Netherlands: ESA contract no. 8903/90/NL/JG
2. Kellner, A.; Schielow, N.; Zapp, F.; MARS: A Generic Mission Planning Tool. In Proc 1st. Symp. on Ground data systems for spacecraft control. Darmstadt, FRG: ESA SP-308, October 1990
3. Ohlendorf, G.; Schielow, N.; NEPTUNE: A new expert planning tool for users in a network environment; Future Generation Computer Systems 7 (1991/1992) 391-401;
4. Calvignac, B.; Assessment of Columbus Payload Operations Capabilities by Scheduling with the MARS System; Round Table on Planning and Scheduling Tools; 6/7.4.1992 ESTEC
5. Pidegon, A.; VEGA; Spacecraft Autonomy Concept Validation by Simulation; Phase 2 Final Report; 28.8.1992

A QUICK UPLINK, EXPANDABLE EXECUTABLE FOR THE NSSC-I OF THE HUBBLE SPACE TELESCOPE

N94-23859

Glenn T. Foley and Elizabeth A. Citrin

NASA/Goddard Space Flight Center
Greenbelt, Maryland 20771

ABSTRACT

The Hubble Space Telescope, launched in April 1990, contains two primary on-board computers. In the past, modifications to the flight software have been accomplished via patches to the on-board executable image (performed while software is executing), or via halt and reload of the computer memory with a new flight software version. This paper describes a method of reloading flight software with a new software version while continuing the on-board execution of a reduced set of spacecraft payload functions.

Key Words: Aerospace, operations, flight software, maintenance

1. Introduction

The Hubble Space Telescope (HST), launched in April 1990, is a 2.4 meter aperture telescope system designed to provide observational capabilities well beyond that of existing ground-based telescopes. This unique orbiting facility is supported by a combination of dedicated and institutionally-provided flight and ground systems which enable science planning and scheduling, mission and science control operations, and data acquisition, processing, analyses, and archival. The HST is planned to operate for at least 15 years.

The HST contains two primary on-board computers. Both computers, designed in the 1970's, are severely memory constrained in relation to current HST flight software requirements, and post-launch experience has necessitated frequent, relatively extensive software modifications. This paper examines the various methods available to install software modifications and discusses, specifically, software modifications relating to

one of the HST on-board computers, the NSSC-I (NASA Standard Spacecraft Computer, Model I).

1.1 HST Observatory

The HST Observatory, the on-orbit portion of the HST, has three major components: The optical telescope assembly (OTA), the Support Systems Module (SSM) and the payload. Figure 1 shows the relationship among these subsystems.

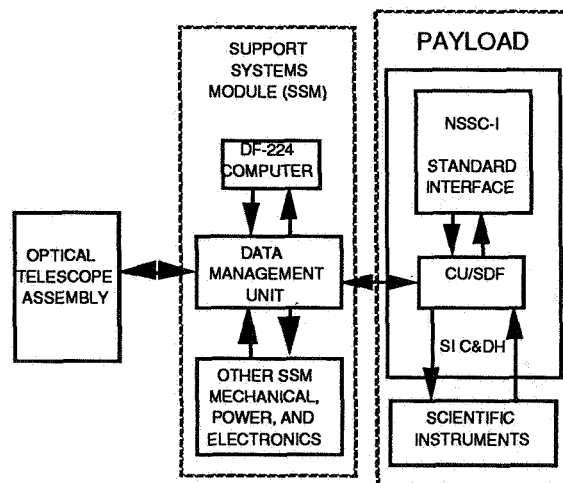


Figure 1. HST Major Subsystems

The OTA is a 2.4-meter Ritchey-Chretien telescope consisting of a primary and a secondary mirror. The telescope's focal plane is divided among four axial science instruments (SIs), one radial SI, and three Fine Guidance Sensors (FGSs). The FGSs perform both pointing control and astrometry functions. The OTA also includes thermal and optics control electronics.

The SSM contains the data management system, including the central computer for the

spacecraft, the DF-224. The SSM provides thermal control, electrical power, communications, data management, and pointing control in support of the OTA, OTA equipment section, the payload, and other SSM subsystems. The SSM data management system interfaces to the payload's Scientific Instrument Control and Data Handling (SI C&DH) module.

In addition to the SI C&DH, the payload has a complement of five scientific instruments that includes two cameras, two spectrographs, and a photometer. Within the SI C&DH, the Control Unit/Science Data Formatter (CU/SDF) and the NSSC-I manage commands and telemetry within the payload.

1.2 NSSC-I Computer

The NSSC-I computer includes a fixed-point arithmetic central processing module and 64K words of random access memory. HST flight software residing in the NSSC-I includes two basic components, the flight executive (Executive) and SI application software. All of the NSSC-I flight software is written in assembly language.

The Executive software is described in detail in the *Multimission Modular Spacecraft On-Board Computer Flight Executive Technical Description* (Ref. 1). This software is responsible for processing commands to the HST payload (initiated from the ground, other NSSC-I software, or stored command memory), and collecting telemetry from the payload and forwarding it to the SSM for transmission to the ground. The Executive performs safety monitoring functions for the payload. The Executive is also responsible for scheduling the execution of other software processors within the NSSC-I, and for performing diagnostic self tests to ensure integrity of the NSSC-I and its software.

The SI application software in the NSSC-I is tailored to the requirements of each science instrument. Generally speaking, this software performs functions such as configuring the SI mechanisms and optics for use, acquisition of astronomical targets, observations with the instrument, health and safety monitoring, and processing and quality checking of science data.

2. Changing the NSSC-I Software

Throughout the early lifetime of the HST mission, desired enhancements to the functionality of the payload control computer have been continually identified, from improved on-board target acquisition algorithms for the science instruments, to increased diagnostic information for daily operations. In addition, as some of the payload hardware has aged, failures and reduced sensitivities have warranted correction in the form of software fixes and workarounds. Some of these more critical changes were implemented as emergency software updates, but most were simply added to a list of desirable new capabilities. As HST settled into normal operations after about a year in orbit, an effort was begun to implement about forty such new desired capabilities. This new product was named the Baseline 4.0 software for the NSSC-I.

When new requirements for the NSSC-I software have been identified, it is necessary to define a method for installing the changes on-board the spacecraft. Ideally, we wish to minimize risk to the health of the spacecraft as well as the dedicated observatory time needed to make the changes become part of the operational system. Several software installation options are available. Selection of a method depends on the quantity of changes being implemented, how the design changes are distributed throughout the system, and how critical the NSSC-I function that they affect is.

2.1 Patching NSSC-I Memory

Because of the extreme cost associated with operating the observatory, it is desirable to install NSSC-I software changes without having to halt the computer and interrupt the timeline of scientific observations. Also, since spacecraft communication contacts are costly, it is beneficial to minimize the actual number of NSSC-I words that must be modified to implement a software change. For example, if a single piece of logic in a program is the only area affected by a change, it is desirable to overlay the new machine language on top of the old code (and spare memory, if necessary),

rather than reloading the entire affected program. This method is referred to as *patching*. Patching can provide an extraordinary savings in uplink time, but often requires very meticulous, bit-level composition of machine language instructions by hand. Needless to say, creating a patch can be a very time consuming and error prone process. In addition, long term maintenance may become an issue as the number of patches that have been performed on a baselined NSSC-I load module grow. Maintaining an accurate map of the pieces of available spare memory is a necessity and reclaiming fragmented space for desired new uses is not always possible. In addition to living with "spaghetti code," the patch versions of changes are inevitably less efficient than inline implementation of logic because of the extra memory words and CPU resources required to transfer control between all of the patch fragments. Some changes, by their very nature, are not patchable because they affect a critical realtime function - execution of that function when the patch is only partly installed could give disastrous results! In any case, the sheer magnitude and complexity of the NSSC-I Baseline 4.0 changes rendered implementation of the changes in patch format to not be a viable option.

2.2 A Hardware Reload of NSSC-I Memory

Given that uplinking Baseline 4.0 in patch format was not feasible, reloading all of NSSC-I memory was the only option available. The standard method for performing this is a *hardware memory load*. In a hardware load, the NSSC-I is halted, and each of the sixteen, 4096-word banks of memory is in turn loaded with its new contents. Though it avoids the difficulties involved in patching, a complete reload provides its own drawbacks. The greatest detractor is that for the period of reloading, the NSSC-I computer must be halted. This is a great impact to spacecraft operations because the science instruments must be commanded to a safe state and the science timeline must be stopped for the duration of the memory load. Also, detailed engineering telemetry describing the health of the payload is not available when the NSSC-I is not running - only a reduced set of fixed telemetry parameters are reported, providing spacecraft operators with reduced visibility into the state

of the payload. Several orbits of spacecraft time are sacrificed for the numerous dedicated contacts required to perform the load. The hardware load process is also not very forgiving: if a single word in the middle of a bank load is dropped due to a communications dropout, the entire 4096-word bank must be reloaded from scratch.

Degradation of some elements in the HST payload has made hardware reloading of the NSSC-I even less desirable. Since the NSSC-I is responsible for the safety of the five Hubble science instruments, the SIs must be pre-commanded to a safe state before the NSSC-I is halted. Unfortunately, two of the SIs have experienced problems which make it desirable to keep them out of their respective safe states.

The Goddard High Resolution Spectrograph (GHRS) suffers from an intermittent power supply failure that prevents it from transmitting its science data, effectively halting its observation. It was discovered that this problem is mitigated by thermally stabilizing the affected area by keeping one of the instrument's main electronics boxes (MEBs) powered on continuously. Unfortunately, this MEB cannot stay on without being monitored by the NSSC-I, and instrument engineers fear that thermal cycling of the failed area may eventually lead to a permanent failure of the faulty hardware. It would therefore be desirable to somehow allow the GHRS MEB to stay powered on for the period of the NSSC-I reload.

Similarly, the Wide Field and Planetary Camera (WFPC) instrument suffers from a contaminant in its optics. After the WFPC's thermo-electric coolers (TECs) drive the instrument's charge-coupled device (CCD) detectors to their cold operating temperature, a costly and time consuming decontamination and recalibration procedure is performed to drive the contaminant from the optics and compensate for its effects. Unfortunately, WFPC's safe state includes having its TECs powered off, which results in the instrument warming up and the contaminants vaporizing, only to recondense on the sensitive WFPC optics when the TECS are later turned back on. This requires the decontamination and recalibration procedure to again be performed. Therefore, it

would be desirable to allow the WFPC TECs to remain on during the NSSC-I reload.

2.3 A Software Reload of NSSC-I Memory

All NSSC-I memory loads that are performed when the NSSC-I is running, whether they are loads of stored command memory, SI observation control tables, or even the previously described software patches, are performed via *software memory load*. In a software load, realtime commands from the ground send up blocks of from one to 62 contiguous NSSC-I memory words and a destination where the words are to be loaded. The block is also tagged with a checksum word that allows retransmission of the block from the ground if the NSSC-I detects an error or data dropout. The unfortunate drawback of software loading is that since the NSSC-I must be running, all of memory could not be loaded using this method because one would overwrite the software performing the memory load with the software being loaded! Clearly, software memory loading on its own will not solve our problem.

3. Elements of the Ideal Reload

Thus far in our examination of patches and hardware and software memory loads, we have seen several elements that would be desirable during performance of our NSSC-I reload. We would like reload all of memory and not have to patch the new functions in, yet we would also like to have the NSSC-I running to provide nominal engineering telemetry visibility into the payload, some level of SI safing protection, and the ability to load memory in small "chunks," with some success/failure feedback to the ground system. In essence, we would like to have a subset of the existing NSSC-I Executive functionality.

Given that we could extract a core of the existing NSSC-I software, supplemented with minimal custom software, the next question would be how we could put this to use in the NSSC-I hardware environment. To address this, let us more closely examine the layout of NSSC-I memory. As discussed earlier, the 65,536 words of NSSC-I memory are divided into sixteen 4096-word banks. The first six of these banks contain program data, the next five

and a half contain the program code, and the remaining four and a half are used for the timeline of stored commands that instruct the payload through its daily operations. In any of the NSSC-I reload approaches considered, standard payload operations would have to be temporarily suspended, even if only for a very short time. Given this inevitability, the stored command memory would not be required through this period. This gave rise to utilization of this memory to house the target reduced set of NSSC-I capabilities required to perform our reload. The software functions desired, together with the operational procedures and contingencies to make use of this software, were labelled the Quick Uplink, Expandable Executable for the NSSC-I, or QUEEN.

3.1 QUEEN Technical Description

A major advantage of the QUEEN approach is that it is composed of a subset of existing software. We shall now examine the QUEEN requirements, first in terms of capabilities of the full NSSC-I Executive, then by describing the software custom written for QUEEN. The design goal for memory utilization was to have QUEEN require less than two NSSC-I banks to operate: 4096 words for code, and 4096 words for program data.

To allow spacecraft operators full visibility into the health of the payload, QUEEN provides for collection of all normal mode engineering telemetry from the various payload elements, including insertion of special QUEEN operational status information.

NSSC-I memory loading and dumping will be performed by inclusion of two of the existing "Executive requests" to software memory load and dump NSSC-I memory. This is the core of the QUEEN concept. Any of the other Executive requests, such as "execute a program" or "dump a data log," are unceremoniously ignored.

QUEEN supports nominal intercomputer communications between the NSSC-I and the DF-224 flight computer through exchange of their processor interface tables (PITs) every half second. QUEEN maintains the NSSC-I clock synchronously with that reported by the

DF-224 in its PIT, and will continue to issue time code update commands to the payload control unit once per minute, as expected.

The QUEEN payload safing capabilities were derived as special cases from the more general safing protection offered by the full NSSC-I Executive. QUEEN safing consists of issuing commands to turn off critical subsystems in either the WFPC or GHRS, or for the entire NSSC-I/WFPC/GHRS complement, in the event of a systemic problem.

Minor modifications were made to the inherited portion of the Executive to allow access to the QUEEN-specific payload safing capabilities. QUEEN payload safing may be invoked by a request from the ground (in the form of a single word memory load to a predefined location), when commanded by the DF-224's PIT, or upon absence of a predefined number of DF-224 PITs. Loss of the "master timing pulse" or one megahertz clock control signals from the payload control unit will also cause QUEEN payload safing. For additional protection, the NSSC-I would signal the DF-224 through the PIT upon detection of a "delayed command error" from the payload control unit. Such an error would indicate a hardware failure that would prevent the NSSC-I from providing proper protection to the payload elements. In this case the DF-224 would command the payload to its safe state until the problem can be resolved.

Finally, QUEEN includes custom software that mimics the more general purpose engineering data limit checking capabilities of the full Executive. This allows on-board inspection of critical payload parameters against specified limits, resulting in safing of a payload element if an item remains out of limits for too long.

3.2 QUEEN Operational Usage

Actual implementation of the QUEEN NSSC-I code was only part of this approach to loading the flight software. Development of procedures to install and utilize QUEEN and prepare for any contingencies was also necessary. The list of contingencies, failure scenarios, and backout procedures is quite extensive - we shall discuss only the nominal

installation procedure. Figure 2 presents the four main steps of QUEEN usage.

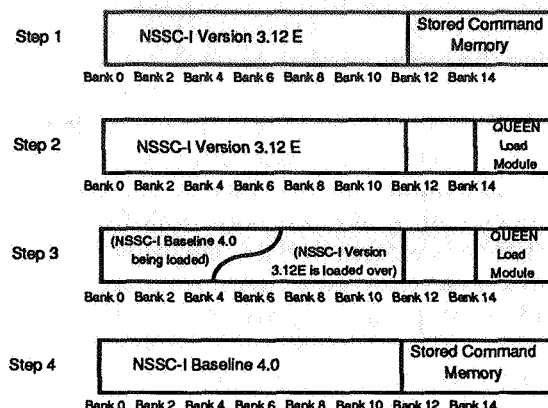


Figure 2. QUEEN Usage

First, with the previous version of the flight software running (Version 3.12E), the stored command timeline is stopped and all SIs except GHRS and WFPC are commanded to their safe states (Step 1). Next, under version 3.12E control, QUEEN is software loaded into the last 2 banks of NSSC-I memory (Step 2). QUEEN is activated by a burst of commands to the NSSC-I hardware which instruct the computer to halt itself, then restart using code from the high memory where QUEEN has been loaded. In Step 3, QUEEN loads Baseline 4.0 over 3.12E into low memory in small pieces that it receives from the ground, as it performs the telemetry collection, intercomputer communications, and safing protection functions described earlier. Finally, when 4.0 loading is complete, another burst of commands is sent to the NSSC-I hardware instructing it to halt, then restart - this time from the low memory to which the new software has been loaded. Under Baseline 4.0 control, stored command memory is reclaimed by loading a fresh supply of stored commands over the memory that QUEEN occupied, and normal operations are resumed (Step 4).

4. Applicability to Other Missions

The concept of a mini-executive for a spacecraft computer is not new to HST. The concept was used on the International Ultraviolet Explorer (IUE) and Solar Maximum Mission (SMM). IUE dedicated one 4K bank as a backup mini-executive to be used in case of problems with

the main 8K executive. Its functionality included telemetry collection and attitude control. It has been used to reload the main executive software. SMM used the concept during the repair mission, to allow software loading of the computer, thus reducing the time required to reload. SAMPEX, the first of the small explorer series of spacecraft, was launched in June of 1992. It includes a 386 processor with 512K of memory. SAMPEX flight code is written in the C language. SAMPEX has adopted a flight software modification philosophy which allows for complete tasks to be replaced. The compiled task is uplinked to a spare area of memory, and appropriate pointers in the assembly language image are updated. This philosophy avoids the timeline disruption of a full reload, while preserving the advantages of fully integrated code changes (except for the task pointers).

It appears that the QUEEN concept is reasonable to pursue for missions with on-board computer software requiring frequent and/or extensive changes where memory is constrained. HST is developing a similar mini-executive for the DF-224, planned to be used in a flight software installation in April of 1993. For future missions where flight software maintenance activity is expected to be high, provision of adequate spare memory would appear to allow for better solutions. Spare memory could be used for a full reload of the flight software image, without interruption to ongoing activities, or a task reload philosophy, like SAMPEX, could be adopted.

5. Conclusions

The issues involved in installing enhancements into the NSSC-I payload control computer of the Hubble Space Telescope have been examined, along with the advantages and disadvantages of the various methods available to achieve this end. QUEEN was an attempt to incorporate the positive aspects of these various methods into a single working product and procedure designed to aid science planning and spacecraft operations by providing nominal payload health and safety protection and telemetry, more efficient use of spacecraft resources, and of course, the desired NSSC-I software upgrades afforded in the new Baseline 4.0 software. QUEEN was successfully

utilized to accomplish the reload of the HST payload computer in June 1992. It is planned to be used again for the NSSC-I Version 5.2, software that will support the first HST servicing mission (scheduled for installation in November 1993).

In a software development sense, QUEEN was a significant and meaningful activity. It embodied such state of the art concepts in software engineering as reusability (with its inherited software core) and total quality management (with the interactive spiral development process of software and procedures among the NSSC-I software engineers, science planners, and spacecraft operators). Yet these current trends were applied to a vintage hardware and assemble language software environment - ultimately yielding a useful and reusable utility.

REFERENCES

1. NASA, Goddard Space Flight Center. *Multimission Modular Spacecraft On-board Computer Flight Executive Technical Description*, S-700-56, July 1982.

N94-23860

GROUND DATA SYSTEM FOR SPACE FLYER UNIT (SFU)

Takahiro Yamada, Susumu Sasaki and Keiken Ninomiya

The Institute of Space and Astronautical Science (ISAS)
Sagamihara, Japan

ABSTRACT

The Space Flyer Unit (SFU) is an unmanned, multi-purpose, retrievable and reusable space platform. The first mission of SFU (SFU-1) will be launched by NASDA's H-II launch vehicle in early 1995, and retrieved by NASA's Space Shuttle after several months in orbit. Two Japanese ground stations, several ground stations of NASA, and a ground station in Chile are used for tracking of SFU-1. The control center of SFU-1 is the Sagamihara Operations Center (SOC) of ISAS located in Sagamihara, Japan. This paper describes the tracking and data acquisition network for SFU-1, the configuration and design policy of the SFU operations control system, and data processing schemes used for mission operations of SFU.

Key Words: Mission operations, spacecraft tracking system, space data system

1. INTRODUCTION

The Space Flyer Unit (SFU) is an unmanned, multi-purpose, retrievable and reusable space platform (Ref. 1). SFU is jointly developed by the following three institutes of the Japanese government: (1) the Institute of Space and Astronautical Science (ISAS) under the Ministry of Education, (2) the Institute for Unmanned Space Experiment Free Flyer (USEF) under the Ministry of International Trade and Industry, and (3) the National Space Development Agency of Japan (NASDA) under the Science and Technology Agency.

The SFU spacecraft consists of the core system and payloads (experiment equipments). The core system provides the payloads with standardized services such as electric power, command and data handling, etc. The core system of SFU is reusable if appropriate refurbishment and maintenance are performed after the flight. Therefore, SFU can make several flights

without major modifications to the core system, each time with a different set of payloads. Each payload is independently operated provided its operations do not cause resource conflicts with other payloads.

The first mission of SFU (SFU-1) will be launched by NASDA's H-II launch vehicle from Tanegashima Space Center in Japan in early 1995, and retrieved by NASA's Space Shuttle after several months in orbit. The operational altitude is about 500km, while the retrieval will take place at about 300km. SFU-1 has the following eleven payloads:

- Two-dimensional array deployment
- High voltage solar array experiment
- Infrared telescope
- Electric propulsion experiment
- Material experiment
- Space biology experiment
- Space plasma measurement
- Gradient heating furnace
- Mirror heating furnace
- Isothermal heating furnace
- Testing of the exposed facility of the Japanese module of the Space Station Freedom

The following sections describe the tracking and data acquisition network for SFU-1, the configuration and design policy of the SFU operations control system, and data processing schemes used for mission operations of SFU.

2. TRACKING AND DATA ACQUISITION
NETWORK FOR SFU

The ground stations to be used for a particular flight of SFU are selected based on the operational requirements of that flight, and the location of the control center is also decided based on the responsibility assignment of that flight.

Fig. 1 shows the tracking and data acquisition network for the first flight of SFU (SFU-1). The primary ground station for SFU-1 is the Kagoshima Space Center (KSC) of ISAS located in the southern part of Japan. The Okinawa Tracking and Data Acquisition Station (OTDS) of NASDA, located in an island in the southern part of Japan, is used as a backup ground station to ISAS KSC during critical phases.

Since the operational altitude of SFU-1 is about 500km, contact with those Japanese stations can be established only during five or six orbits out of 16 orbits per day. Therefore, three Deep Space Network (DSN) stations of NASA (i.e. Goldstone, Canberra and Madrid) and three Ground Spacecraft Tracking and Data Network (GSTDN) stations of NASA (i.e. Bermuda, Wallops and Merritt Island) are used to extend the ground coverage during critical phases. The Santiago ground station of the University of Chile is also used during critical phases. Since the Santiago station covers almost the same orbits as the Japanese stations augmenting the coverage from the Japanese stations, the Santiago station serves as an ideal auxiliary station when critical operations are performed over the Japanese stations.

The control center of SFU-1 is the Sagami-hara Operations Center (SOC) of ISAS, which is in the city of Sagami-hara of Japan. All flight operations for SFU-1 are performed at SOC except for the proximity operations phase during which SFU communicates directly with the Space Shuttle Orbiter.

Except for the proximity operations phase, all commands are generated at SOC and transmitted to SFU through a ground station in realtime. Realtime telemetry data received at a ground station is transferred to SOC in realtime, where it is monitored, analyzed and archived. Playback data from the onboard data recorder is temporarily stored at the receiving ground station and delivered to SOC in an offline mode after the tracking pass.

Orbit determination is performed at SOC and the Jet Propulsion Laboratory (JPL) of NASA during critical phases. During the mission operations phase, orbit determination is performed at the Tracking and Control Center (TACC) of NASDA based on the RARR data from ISAS KSC. During the rendezvous operations before the retrieval, C-band radar stations of the U.S. Government are used for orbit determination of SFU, and the Johnson

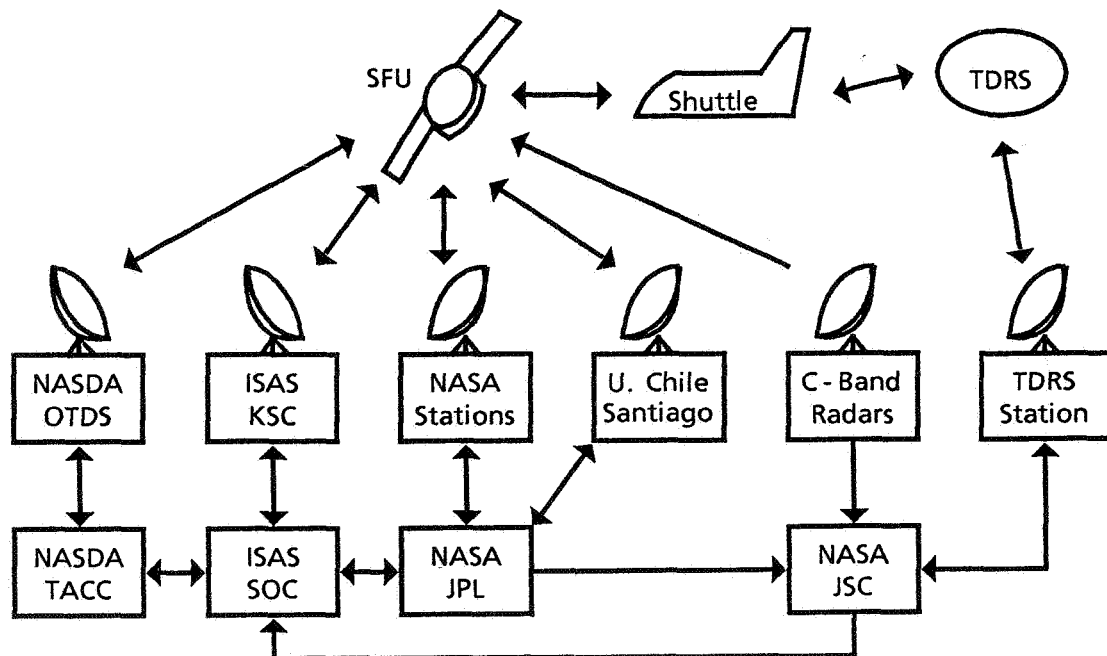


Fig. 1 Tracking and Data Acquisition Network for SFU-1

Space center (JSC) of NASA provides SOC with the state vectors of SFU.

During the proximity operations phase, telemetry data is monitored at the Space Shuttle Orbiter, the Mission Control Center (MCC) of JSC and SOC. Commands will probably be issued at the Space Shuttle Orbiter and MCC of JSC during this phase.

3. SFU OPERATIONS CONTROL SYSTEM

3.1 Design Philosophy

The SFU operations control system is dedicated to SFU, not shared with other projects, because the mission operations requirements for SFU are very different from those of other Japanese space missions. The following is the design philosophy of the SFU operations control system:

- 1) Virtually any ground station can be incorporated into the system with only a small modification to the gateway software. The control center can be placed at any of the three institutes participating in the SFU project.
- 2) A distributed system architecture and a modularized software structure are adopted so that the system can be modified easily for future flights of SFU and new technology can be introduced without modifying the entire system.
- 3) Industry standard software and communications protocols are used whenever possible in order to facilitate interconnection of computers of different manufacturers.

3.2 System Configuration

Fig. 2 shows the basic configuration of the operations control system for SFU-1. There are some other offline computers which are not shown in Fig. 2. All the computers at SOC use the UNIX operating system, and most of the software developed for this system is implemented in the C programming language.

There are two major local area networks (LAN's) at SOC, which are shown as two thick lines in Fig. 2. These are standard

Ethernets, and TCP/IP is used as the communications protocol. The left LAN (SFU LAN) is basically for online spacecraft operations, while the right LAN (SFU Experiment LAN) is used for distributing data to experiment data processing computers.

The SFU Gateway is a communications processor used to communicate with ground stations and other space centers such as JPL and JSC. The SFU Gateway receives telemetry data, orbit data (state vectors and RARR data) and administrative data (sequence of events, etc.) from a ground station or another space center, and transfers the received data to an appropriate computer at SOC with the SFU standard protocol and format. The SFU Gateway receives command data, orbit data and administrative data from a computer at SOC, and transfers the received data to an appropriate ground station or center. The SFU Gateway also stores the received data temporarily in case the data is lost during transmission. The SFU gateway consists of a primary computer and a backup computer.

The SFU Monitor and Control Computer (SMCC) is used to issue commands and to monitor and archive housekeeping telemetry data. SMCC consists of a primary host computer, a backup host computer and several workstations for man/machine interface. These workstations are connected to one of the host computers via a LAN local to SMCC.

The Timeline Generator (TLG) validates planned operations sequences and generates command sequences. TLG consists of a primary workstation and a backup workstation.

The Experiment Monitor Computer (EMC) is used to monitor and archive experiment and housekeeping telemetry data. It distributes telemetry data with some ancillary data to the Experiment Data Processing computers where offline analysis of experiment data is performed. EMC consists of a host computer and several workstations for man/machine interface. These workstations are connected to the host computer via a LAN local to EMC.

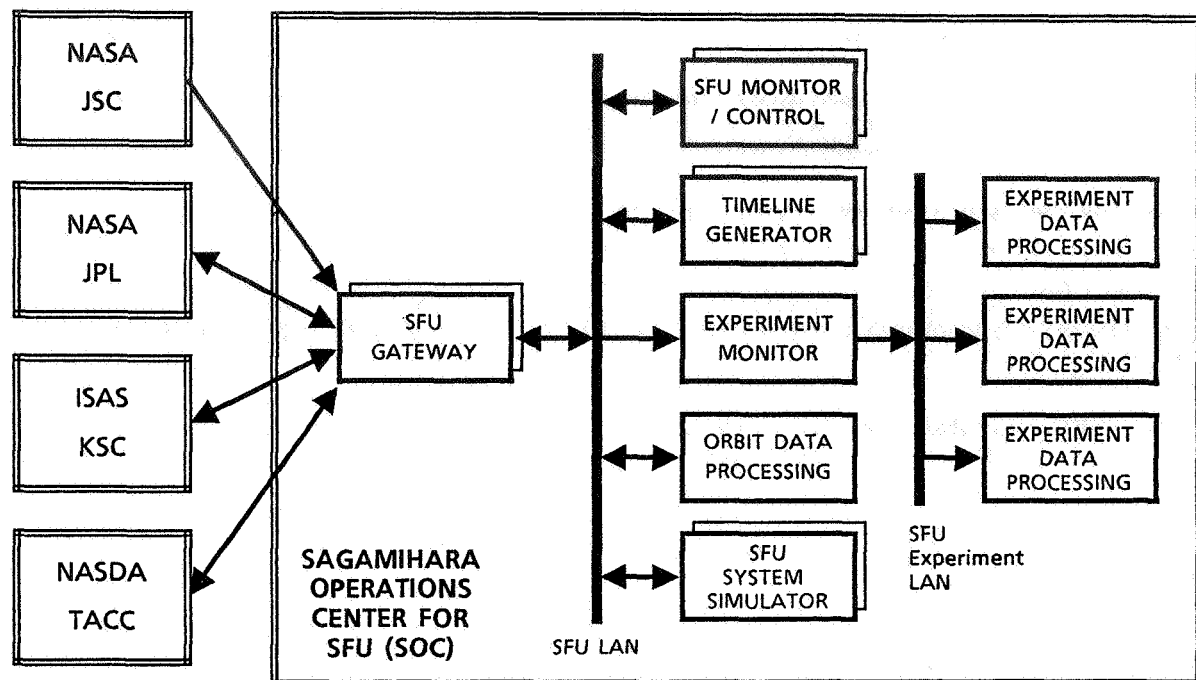


Fig. 2 SFU-1 Operations Control System

The Orbit Data Processing Computer (ODPC) performs orbit propagation and coordinate system conversion. ODPC consists of one workstation.

The SFU System Simulator (SSS) receives commands from SMCC, simulates some of the subsystems and experiments of SFU, and sends telemetry to SMCC and EMC. SSS consists of two host computers (primary and backup) and two workstations (primary and backup).

The experimenters can bring any computers as Experiment Data Processing Computers (EDPC's) if they can be connected to the SFU Experiment LAN. For the first flight, several minicomputers and several workstations will be used as EDPC's. There is no connection to other computer networks such as Internet because of security requirements.

4. COMMAND DATA PROCESSING

4.1 Mission Planning and Command Generation

Mission planning is carried out by integrating into the mission timeline the Functional Objectives (FO's) requested by the experimenters and flight planning

team. An FO is a unit for generating the mission timeline. Each FO has a sequence of commands to be executed, the amount of resources required to perform the FO, and FO execution rules related to that FO (e.g. FO's which must be executed prior to that FO). There is a standard format to describe FO's and each FO is generated as an FO file by some offline computer. The command sequence in the FO file is written in a high level language called the SFU Command Language (SCL). Some FO files (e.g., FO's for orbit correction maneuvers) are automatically generated by an offline computer.

The requested FO's are arranged by an operator, and a time ordered FO sequence is fed to the Timeline Generator (TLG). TLG checks whether or not the FO sequence violates resource allocation rules or FO execution rules. If the sequence violates any of the rules, an operator modifies the FO sequence through interactive operations with TLG. When this process is completed and the final FO sequence is obtained, TLG generates an Event File which contains the integrated sequence of commands generated from the original FO files. TLG also calculates event times such as AOS and LOS times, and includes the event information in the

Event File. The entire Event File is written in SCL.

4.2 Command Execution and Verification

The Event File generated by TLG is sent to SMCC (Fig. 3). The Event File is then interpreted and executed in the following way. Commands to be executed during a station contact are converted to binary command data and transmitted to the spacecraft with some interactive operations by the operator. Commands to be loaded in the onboard command memory and executed later are converted to memory load commands and transmitted to the spacecraft by the directions of the operator. Commands in the Event File are automatically transmitted unless other operations are specified in the Event File. The operator can abort the execution of the Event File and manually issue commands any time.

If the spacecraft must be in a certain status when a command is going to be executed, then SMCC verifies the status of the spacecraft using the received telemetry data before sending the command. After sending a command, the result of the

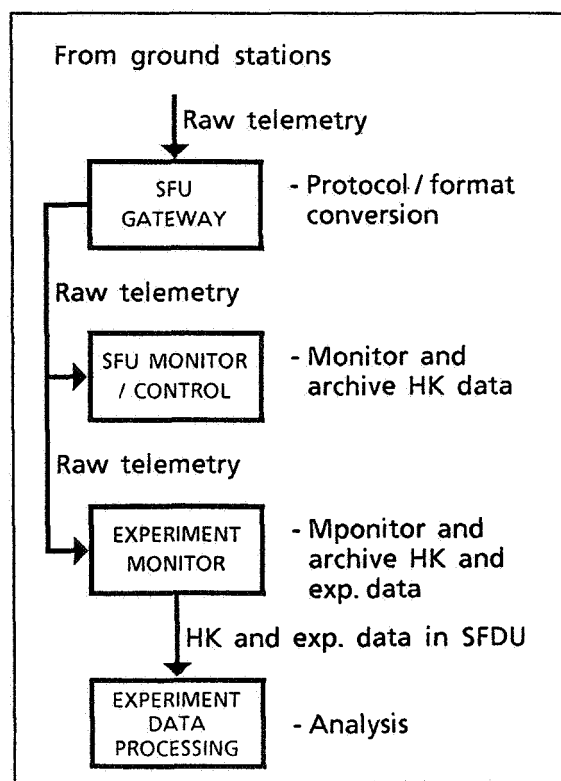
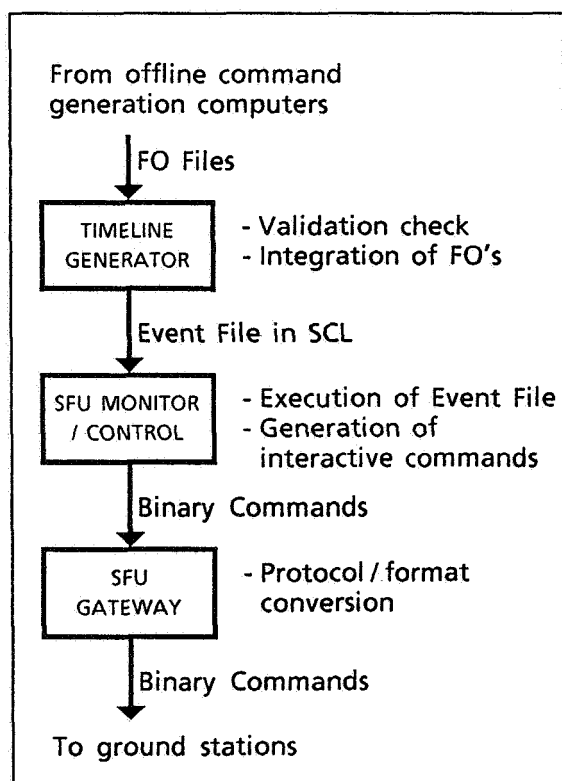
command execution is verified with the telemetry data. When a critical command has been sent, the command transmission process is suspended until the execution result is verified. For a non-critical command, the execution result is not verified right after the transmission. At the time specified in the Event File, the execution results of non-critical commands which have not been verified are verified simultaneously. If a verification fails, SMCC generates an anomaly report, alarm the operator, and abort the execution of the Event File.

5. TELEMETRY DATA PROCESSING

5.1 Telemetry Checking

Received telemetry data is fed to SMCC and EMC simultaneously (Fig. 4). SMCC processes only housekeeping data, while EMC can process both housekeeping and experiment data. Against received telemetry data, two kinds of checking are performed as follows:

- 1) Limit checking - Checking that the value of a received telemetry data is between a pair of fixed limit values.



This checking is performed to detect a hardware error. Limit checking is performed against all realtime telemetry at the time of reception. The limit values can be changed by changing limit data files.

- 2) Mode checking - Checking that the value or status of a set of received housekeeping telemetry data satisfies the conditions described in a specified mode check file. This checking is performed to check that the spacecraft is actually in the desired mode. SMCC automatically executes mode checking if it is described in the Event File. The operator can manually initiate mode checking any time. Mode checking can also be performed in a batch mode against the telemetry data of a specified time archived in SMCC. Batch mode checking is useful for verifying the mode transitions of the spacecraft during an out-of-contact period.

If a check fails, an anomaly report is generated and the operator is alarmed to take an action.

5.2 Telemetry Display

Realtime telemetry data is monitored in realtime on the workstations of SMCC and EMC. Archived data is also displayed with the directions of the operator on these workstations. The display format is defined with a window definition language called the Display Format Descriptor (DFD). SMCC and EMC reads the display format request defined with DFD from a file, and generates the display.

Several windows can be drawn on one workstation, and a window consists of a alphanumeric table or a graphical plot. Some of the windows can be placed on the screen as icons, which can be opened with a direction of the operator. The display format request contains the following information for each window: the data items to be displayed in the window, the format of the table or plot, the colors to be used for normal data and out-of-limit data,

etc. The result of some arithmetic operations can also be displayed by defining the operations with DFD. If a data fails the limit check, the color of the data on the display changes. If the data is being displayed in one of the icons, the color of the icon changes.

5.3 Telemetry Distribution

SMCC can provide a text file called the SFU Parameter File which contains the value of all housekeeping telemetry items derived from one major frame of telemetry. This file is used by TLG and SSS as the initial condition of timeline generation or simulation. EMC provides EDPC with the set of telemetry data requested by an experimenter in the SFDU format.

6. CONCLUSIONS

This paper described the tracking and data acquisition network and the operations control system of SFU-1. The operations control system is now in the stage of integration testing which will be finished in the fall of 1993.

7. ACKNOWLEDGEMENTS

The authors would like to acknowledge the contribution of the people of USEF, NASDA, NASA, University of Chile, Fujitsu, NEC, Hitachi, NED and Melco who have been involved in the development of the SFU ground data system.

8. REFERENCES

1. Kuriki, K. et al. 1990. The design and orbital operation of Space Flyer Unit. In *Proc. 41st Congress of the International Astronautical Federation*: IAF-90-055.
2. Yamada, Takahiro et al. 1992. Communication and data handling system of the Space Flyer Unit (SFU). In *International Space Year in the Pacific Basin*, Volume 77, *Advances in the Astronautical Sciences*, American Astronautical Society.

RADIOASTRON FLIGHT OPERATIONS

N 9 4 - 2 3 8 6 1

V. I. Altunin
Russian Academy of Sciences
Moscow, Russia

K. G. Sukhanov
Babakin Center
Moscow, Russia

K. R. Altunin
Jet Propulsion Laboratory
Pasadena, California

1.0 INTRODUCTION

Radioastron is a space very-long-baseline interferometry (VLBI) mission to be operational in the mid-90s. The spacecraft and space radio telescope (SRT) will be designed, manufactured, and launched by the Russians. The United States is constructing a DSN subnet to be used in conjunction with a Russian subnet for Radioastron SRT science data acquisition, phase link, and spacecraft and science payload health monitoring. Command and control will be performed from a Russian tracking facility.

In addition to the flight element, the network of ground radio telescopes which will be performing co-observations with the space telescope are essential to the mission. Observatories in 39 locations around the world are expected to participate in the mission.

Some aspects of the mission that have helped shaped the flight operations concept are: separate radio channels will be provided for spacecraft operations and for phase link and science data acquisition; 80-90% of the spacecraft operational time will be spent in an autonomous mode; and, mission scheduling must take into account not only spacecraft and science payload constraints, but tracking station and ground observatory availability as well.

This paper will describe the flight operations system design for translating the Radioastron science program into spacecraft executed events. Planning for in-orbit checkout and contingency response will be discussed, also.

2.0 FLIGHT ELEMENTS

The Radioastron flight elements, shown on Figure 1, include the Spectr-R spacecraft and the space radio telescope. The spacecraft will operate in a highly elliptical orbit with an apogee of 80,000 km, perigee of 2000 km, and a 28-hour orbital period.

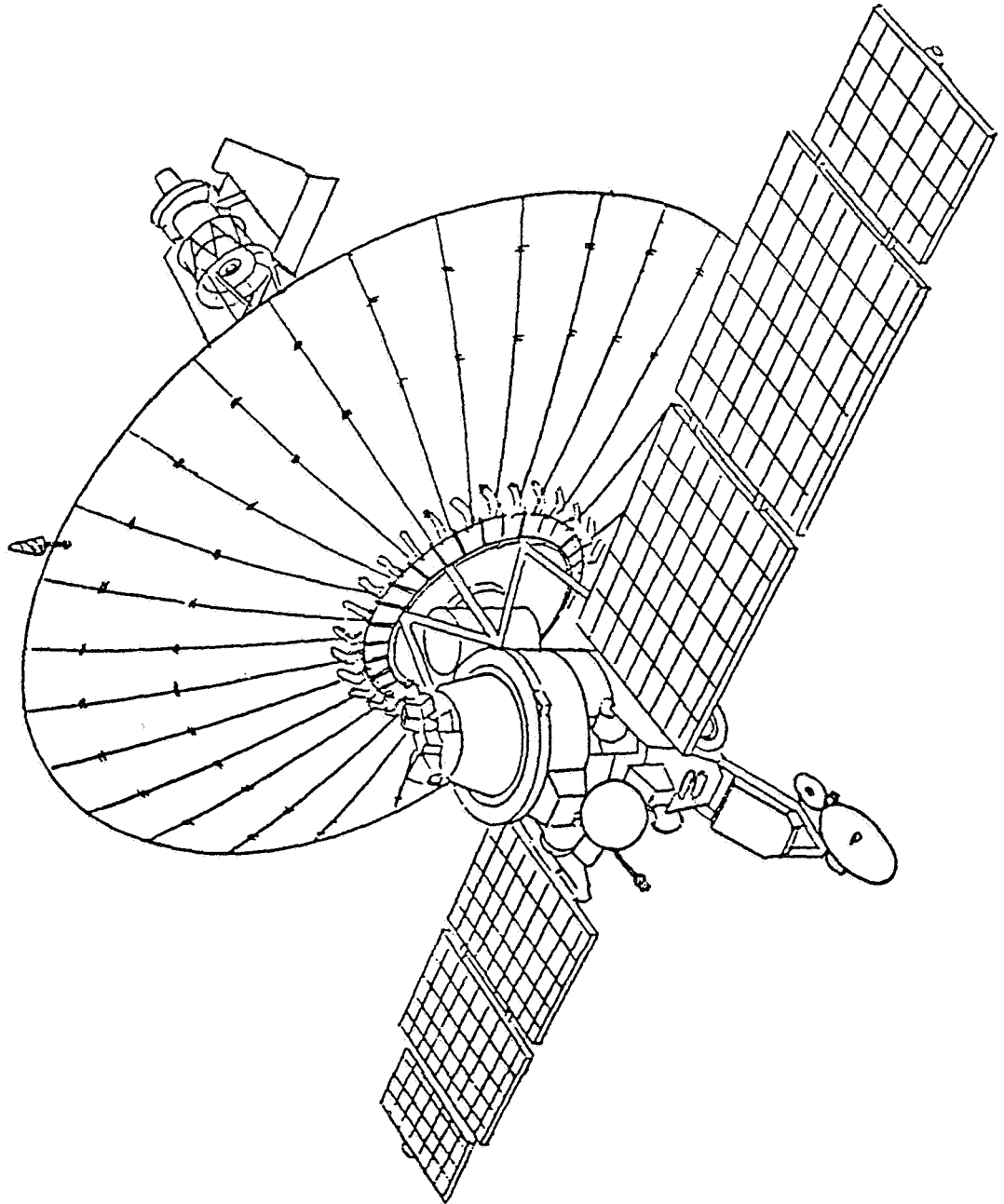
2.1 Spectr-R Spacecraft

The Radioastron spacecraft is part of the Spectr series of spacecraft which was designed for astrophysics research by the Russian space industry in response to a request by the Russian Academy of Sciences. The Spectr bus will be common to three separate astrophysics missions - Spectr-Radioastron, Spectr-X Gamma (for X- and gamma-ray research) and Spectr-UV (for research in the ultra-violet portion of the spectrum). The spacecraft bus design is based upon the designs of previous Russian spacecraft. The 3 spacecraft all share, among other things, the same radio commanding and telemetry systems (Main Radio Complex, MRC) which will operate in C- and P-band, a central processor with onboard memory (Time Program System, TPS), power supply system, a 3-axis stabilized attitude control system which provides pointing accuracy of up to 1 arc minute, solar panel orientation system, and propulsion system. All of these critical systems are redundant except for the propulsion system.

2.1.1 Spacecraft Commanding

The spacecraft will be operated through the MRC. Commands will be transmitted from the ground at a maximum of 10bps through the radio system which receives and decodes the commands. Thereafter, they

FIGURE 1
RADIOASTRON SPACECRAFT



will be distributed to the TPS. The basic function of the TPS is that of a dispatcher of spacecraft and science payload operations that are controlled (activated, deactivated, mode selected) by the corresponding TPS programs.

TPS programs may be started and stopped by functional or numerical commands transmitted from the ground during the daily communication session. The entire memory of the TPS is 10KB which is divided into two areas: flight software and flexible programs.

2.1.2 Spacecraft Telemetry

The telemetry (TM) system is a software-hardware package that is incorporated into the MRC and is used for acquisition, conversion and storage of telemetric information from the spacecraft and science payload. The TM information data rate is 1 Kbps. Approximately 40 MB of TM data can be stored onboard. The data will be acquired in real time or dumped from the recorder using the C- or P-band channel during the daily communication session.

2.2 Space Radio Telescope

The Radioastron science payload consists of a high precision 10m-diameter deployable antenna and four radio-astronomical receivers working at the following wavelengths: 92cm, 18cm, 6cm, and 1.35cm with dual circular polarization. Any two of the receivers can operate simultaneously at the same polarization or one receiver can be operated in the dual polarization mode.

The science payload includes 8 subsystems which are controlled through the C- and P-band radio complex and TPS during the communication session. Commands either can be stored in the TPS or transmitted in real time. Information about the health of the science payload is transmitted to Earth through the TM system and C- and P-band radio complex.

3.0 RADIOASTRON TRACKING

The tracking facilities to be used for Radioastron are one of the primary elements of the ground support segment of the space VLBI mission [1] as are essential to spacecraft and science payload operations.

3.1 Radio Channels

The Spectr-R spacecraft will have four radio channels to communicate with ground support systems. Commanding, TM monitoring of the spacecraft and science payload, and navigation measurements (range and range rate) will be performed using the C-band channel (with P-band as a backup).

The signal from the radio astronomical receivers will be transmitted to Earth through the High Information Rate Radio Complex (HIRC). It will be digitized, formatted, and transmitted by a Ku-band transmitter. The maximum science data rate is 144 Mbps. To support the SRT as part of an Earth-space interferometer, a high-stability signal from a ground-based hydrogen maser standard is transmitted to the spacecraft through the X-band radio link. The spacecraft, in turn, will retransmit the signal to Earth. The signal will be used for navigation measurements (Doppler) and for phase residual measurements which will be used to process the Earth-space interferometric data. The C-band channel can be used as a phase link backup when not in use for satellite control.

The spacecraft control session using C-band can take up to 4 hours during the 28-hour orbital period. The separation of the spacecraft control and science data radio channels provides the possibility to do science observations during a control session (although this is not recommended due to possible interference effects). This could increase the science return of the mission, especially during the in-orbit checkout period when the control sessions are expected to occupy a majority of the operational period. Another advantage to separating the channels is that the C-band channel can then be used as a backup for the X-band phase link.

The DSN subnet to be used for science data acquisition and phase link will be equipped to work only at X- and Ku-bands, not C-band. Thus, the spacecraft can be controlled only during times it is tracked by Russian tracking facilities. Because the spacecraft will be operating autonomously 80-90% of the time, it was decided for contingency recognition purposes that spacecraft and science payload health data should be

included in the science data frame header which will be acquired by the tracking stations together with the high-rate science data stream through the Ku-band radio channel.

3.2 Tracking Facilities for Spacecraft Operations

Radioastron will be operated in Russia from the Satellite Control Center (SCC) through the tracking network located on the territory of the former USSR in Ussuriisk and Evpatoria (both are 32m antennas working in P- and C-bands) and near Moscow (25m antenna working in P-band). The SCC connects with the tracking stations both by telephone (2.4 Kbps) and satellite communication lines (64 Kbps). The same tracking station network will receive TM from the spacecraft and science payload and send this information to the SCC for analysis by the spacecraft and science payload analysis teams. The navigation measurements at C-band (range and range rate) also will be executed by the tracking stations and the results forwarded to the Navigation Centers located in Moscow.

3.3 Tracking Facility for Phase Link and Science Data Acquisition

To optimize the science return and to provide continuous coverage of the space element, a world-wide tracking network is being constructed to support Radioastron. This network will include one Russian tracking station located in the Russian Far East (Ussuriisk) and three US DSN 11m tracking stations (Goldstone, Madrid, Canberra) and a 14m antenna in Green Bank. The US Space VLBI Office at JPL will be responsible for managing the US-sponsored portion of the tracking network (which will also support operations of a Japanese space VLBI mission - VSOP) and the Astro Space Center of the Russian Academy of Sciences in Moscow will be responsible for the Russian portion.

The science data flow (144 Mbps) will be recorded at the tracking stations as well as phase residual measurements (30 Kb per day) and sent to the Data Processing Facility (DPF). There will be one each in Russia and the US. The DPFs will process data from the SRT and the co-observing ground telescopes. The X-band Doppler

measurements from the tracking stations (30 Kb per day) will be sent to the Orbit Determination Centers (ODCs) - one each in Moscow and Pasadena to be used together with C-band range and range rate measurements for orbit predictions and reconstruction. The orbit reconstruction data, required for the correlation process, will be based on about two weeks of measurements. The orbit predictions (Doppler, state vectors) will be sent daily to the SCC and tracking facilities to support operations.

Two of the tracking stations (Ussuriisk and Green Bank) will be equipped with diagnostic correlators to provide near-real time end-to-end system verification of the Earth-space interferometer. These two stations were chosen because they are co-located with ground radio telescopes. At these stations, the high-rate science data from the space element and the science data from the ground telescopes can be processed a few minutes after the observations. The results will be sent to the General Operative Control Group (GOCG) and Radioastron Science Operations Group (RSOG) to show the status of the interferometer. Using the diagnostic correlator will be very important during the in-orbit checkout phase.

4.0 FLIGHT OPERATIONS

The mission operations concept for the Earth-space VLBI mission was described in [2]. The following focuses on the flight operations element.

4.1 In Orbit Checkout

Routine operations for the Radioastron spacecraft and SRT will begin after a 3-month in-orbit checkout (IOC) period. During IOC, the critical spacecraft systems and science payload will be fully tested. Some science data will be collected during this period.

The IOC period will be divided into three phases:

(1) The first 2 -4 weeks will be dedicated to checking the spacecraft systems - the Main Radio Complex, Attitude Control and Power Supply systems.

(2) The second phase will consist of a checkout of the science payload in conjunction with the spacecraft. The Attitude Control System will be retested after the SRT has been deployed because the mass distribution of the spacecraft will change due to the size of the antenna (10m) and mass (800 kg) which is about 20% of the mass of the spacecraft. The attitude control system can be calibrated in parallel with boresighting of the SRT. The SRT characteristics will be checked simultaneously with the thermal control system of the spacecraft.

(3) The final phase will consist of the end-to-end checkout of the Earth-space interferometer by using a diagnostic correlator.

During IOC, spacecraft and science payloads in Russia usually are monitored autonomously by the GOCG. In the case of Radioastron, however, the end-to-end testing of the space elements together with the ground elements is of extreme importance because of the unique relationship between the two elements in an Earth-space interferometer experiment. Therefore, operations during IOC will be executed in real time.

4.2 Routine Operations

The orbital parameters of the spacecraft and the mission planning cycles will govern, to a large extent, routine flight operations for Radioastron. The mission planning cycles for Radioastron are based upon usual ground VLBI and Russian spacecraft operations practices: (1) The minimum amount of time required to change a spacecraft science observing command file is 2 - 3 days; (2) The observing program for Russian space science missions usually covers a month's worth of observation sources; (3) Time commitments for co-observations with ground radio telescopes usually are made one year in advance; (4) Announcements of Opportunity (AO) are usually released one year in advance of the observations covered by the particular AO.

The GOCG, resident at the SCC, will be responsible for preparing the spacecraft command sequences based upon the observation requests generated by the observers and RSOG. The RSOG will

create the science payload command sequences. The command sequences will be generated with the assistance of software which identifies activities in violation of spacecraft and science payload constraints. The GOCG will process the sequences through a hardware and software command simulator prior to transmitting the command files to the tracking station where they will be stored for 2 -3 days before being uplinked.

After the commands have been uplinked, the GOCG will verify that the commands have been properly loaded on the spacecraft and payload. During this time, also, the onboard TM recorder will be off-loaded and real-time TM will be received.

4.3 Contingency Response

During the analysis of the TM data, if spacecraft or science payload anomalies are recognized, the spacecraft and science payload system experts will attempt to diagnose and resolve the problem. They will follow an established set of procedures for identifying the source of the onboard anomaly. The first source to be checked is the tracking system. This is followed by the main radio complex, power supply system, and finally the attitude control system. In prior missions, when an attitude control problem has arisen, it has taken about 2 hours to deduce that it is the source of the anomalous condition. The length of the response time is due to the reluctance to meddle with the attitude control system. Given that time is of the essence when dealing with an attitude control anomaly, the Russian spacecraft team is hoping to generate a new contingency response "tree" to accelerate the identification of the source of the anomaly.

During times when the US-sponsored tracking stations are tracking the spacecraft, they will record and send to the Project Operations Control Center at JPL in near real time, a certain number of blocks of science header data per second. As mentioned earlier, this data will include information on the spacecraft and science payload health. Alarm limit software at JPL will enable GOCG's spacecraft team to be alerted if an anomaly has occurred.

5.0 CONCLUSIONS

The complexity of an Earth-space interferometry mission results in large part from the need to coordinate many disparate elements. During the design stage of the Project, the decision was made to improve the flight control loop of the mission through, for instance, the inclusion of spacecraft TM in the science data header and the use of a diagnostic correlator. The flight operations concept for Radioastron is a unique one, reflecting the uniqueness of an Earth-space interferometric system.

6.0 REFERENCES

1. Altunin, Valery I. 1990. Technical Parameters of the Ground Support Segment and Data Management of The Radioastron Project. *The 2nd International Symposium on Space Information Systems*. Pasadena, CA.
2. Altunin, Valery I., and Robinett, K. H. 1992. Mission Operations System for Russian Space Very-Long-Baseline Interferometry Mission. *43rd Congress of the International Astronautical Federation*. Washington, D. C: IAF-92-0547.

N 9 4 - 2 3 8 6 2

THE ULYSSES SPACECRAFT CONTROL AND MONITORING CONCEPTS AND REALITIES

Paul Hamer and Nigel Angold

Paul Hamer - Cray Systems, Transom House, Victoria Street,
Bristol, England.

Nigel Angold - European Space Agency (ESA), European Space Operation
Centre (ESOC), Darmstadt, Germany.

ABSTRACT

Ulysses is a joint ESA-NASA mission the primary purpose of which is to make scientific measurements of the Sun outside the plane of the ecliptic.

The delay in launching Ulysses, due to the Challenger disaster, meant that the hardware on which the Spacecraft Control and Monitoring System (SCMS) resides was becoming obsolete and it was decided to convert SCMS to run on a DEC/VAX machine under VMS.

The baseline for the conversion was an exact copy of the existing ModComp system from a functional point of view. However, following conversion and installation at the Jet Propulsion Laboratory (JPL) in Pasadena, California, further requirements were identified by the spacecraft control team and this led to the baseline being substantially upgraded to support the mission fully.

The paper will cover the spacecraft, the conversion, the converted SCMS, problems found and the upgrades implemented for solutions. It will also discuss the future for and enhancements already made to the converted SCMS.

Key Words: Ulysses, spacecraft operations, mission control.

1. INTRODUCTION

The Ulysses spacecraft and mission is divided as follows. ESA provides the spacecraft, half of the

science instruments and the spacecraft-dependent mission operations personnel, hardware and software. NASA provides the remaining science instruments, the radioisotope thermal generator (which powers the spacecraft), the launch system and the tracking, telemetry data acquisition and command support via NASA's Deep Space Network (DSN).

The Ulysses spacecraft has a scientific payload of nine experiments and is spin-stabilised, rotating at approximately 5 rpm. It has low gain antennas used in the early stages of the mission and a high gain antenna capable of X and S band transmissions. The spacecraft also has a 72 metre Radial Dipole antenna and a 7 metre axial monopole antenna, both for experiment use.

There is an onboard storage capacity for periods of no ground station coverage, i.e. 2 magnetic tape recorders that can each hold 22 hours of data at the nominal record rate of 512 bps.

Telemetry is available in two formats, Engineering and Science and can be transmitted at rates of between 64 bps and 8192 bps. The higher rates are used to transmit scientific formats from the onboard recorders interleaved with realtime data. The nominal realtime bit rate during routine operations is 1024 bps in science format with 512 bps during the recording periods.

The original SCMS software was developed by Burd Voor Systeem Montwikkeling of Holland (BSO) for ESA to meet the launch date of May 1986. The rescheduling of the Ulysses mission meant that the ModComp hardware would be-

come obsolete during the four year delay and as a consequence SCMS was converted by Cray Systems (formally MARCOL) for ESA to run on a DEC/VAX under VMS keeping FORTRAN as the language used, albeit a different dialect.

2. THE CONVERSION

The conversion process itself could not be performed automatically using a Code Conversion program since the hardware architecture and the low level software interfaces to them were significantly different between the ModComp and VAX machines. This meant that all of the SCMS lower level subsystems had to be completely rewritten. These lower level tasks included the interprocess communication, file management and the terminal interface.

The software development followed the ESA BSSC life-cycle (Ref. 1) which meant the Functional Requirements Document produced for the ModComp system had to be rewritten as a Software Requirements Document (SRD). Within the Software Requirement (SR) phase any new requirements raised on the existing system considered essential were included in the document for review.

As mentioned above the differences in the Architectural Design (AD) of the system were due to hardware differences so the functionality of the design remained the same.

The Detailed Design (DD) and Coding stages saw the development follow the standard approach of reviewing the design prior to implementation.

The Software Transfer (TR) phase proceeded very smoothly resulting in a delivery of the first version of SCMS at ESOC some three months ahead of schedule. This was particularly important as no simulator was available so data recep-

tion from the spacecraft itself was regarded as essential for thorough testing and operations training. A paper written by Dr. Richard Corkill (Ref. 2), the Cray Systems project manager for the conversion, details the conversion process.

3. THE CONVERTED SCMS

The converted Spacecraft Control and Monitoring System (SCMS) resides on a DEC/VAX 8250. Two of these machines make up the Ulysses Mission Control System (UMCS) and are installed at NASA's Jet Propulsion Laboratory (JPL).

Interestingly, SCMS although originally designed over 8 years ago, contains many of the facilities associated with more modern spacecraft control and monitoring systems. Of special interest to SCMS are:

- *Databases*
- *Telemetry Processing*
- *Telecommanding*
- *Archiving of Telemetry*
- *Display of Telemetry*

The SCMS is comprised of a number of intercommunicating tasks. Prioritisation is used to ensure that realtime tasks, i.e. those driven by incoming telemetry get preferential use of the CPU. The relationships between the processing and data can be seen in Figure 1 - SCMS Major Architecture

3.1 Databases

Most SCMS subsystems rely on the contents of one or more databases for their operation. Each database can be accessed via the database editors to update information contained for displays, modes, parameters, limits and commands.

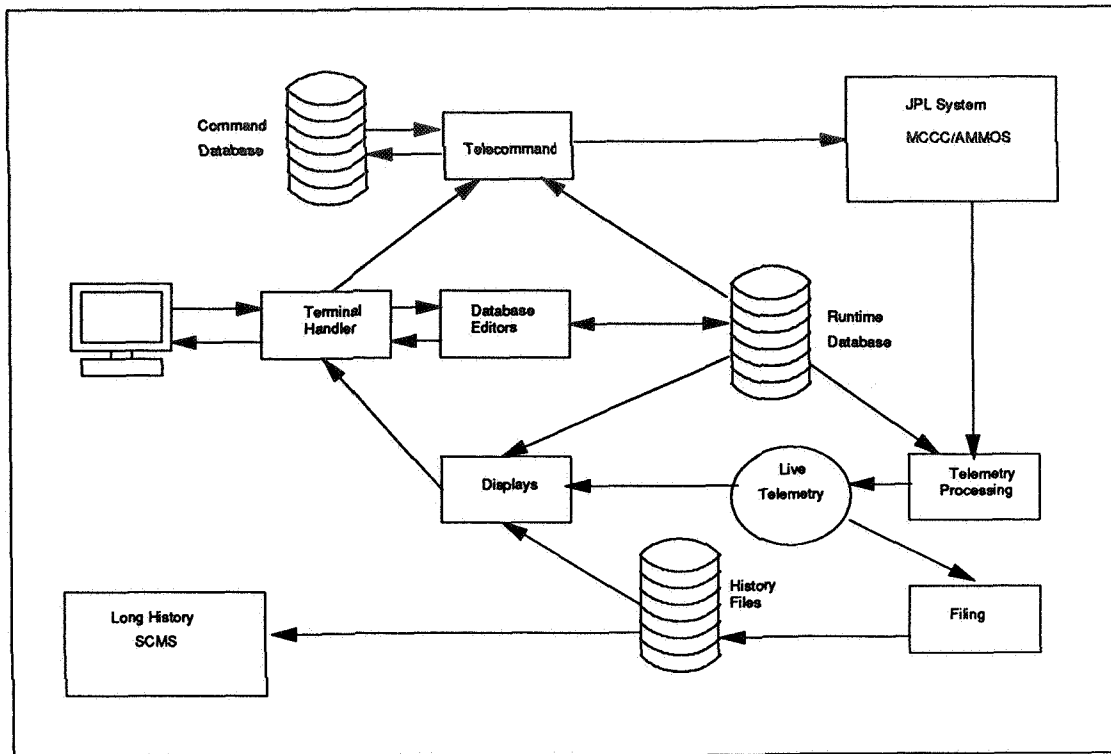


Figure 1 - SCMS Major Architecture

3.2 Telemetry Processing

Telemetry is received from the JPL Ground Data System (GDS) which, as of time of writing is the Data Capture and Staging system (DACS). This is part of the Mission Control and Computer Centre (MCCC) but will be replaced by the Advanced Multi Mission Operating System (AMMOS).

The GDS sends data in the form of Standard Format Data Units (SFDUs) which consist of a header and a 128 byte frame containing spacecraft telemetry. These frames are collected into the number of frames required for a complete telemetry format, which depends upon the type of data : 2 for engineering and 32 for science.

The data can be received in two ways, either realtime or recalled and the data itself can consist of either realtime or playback data.

- *Realtime Data:* This is data received directly from the spacecraft and must be processed immediately.

- *Playback Data:* This is data recorded on-board and is received interleaved with the realtime data to be processed later.

Each telemetry format is processed to produce a processed telemetry record (PTR) which is made available for display and archiving. The record consists of certain portions of raw telemetry associated with platform housekeeping and some 800 derived parameters for spacecraft and scientific data.

3.3 Telecommanding

The way in which telecommanding for Ulysses is performed is one area which differs from most modern systems. This is because commands are ultimately sent via the JPL MCCC system which has no physical link to the SCMS hardware where the commands are prepared. Compare this to modern systems where command preparation and delivery are normally performed on the same machine.

3.4 Archiving of Telemetry

This is achieved by using fixed size circular files, called Short History Files, that contain the PTRs for the last 15 days, an out of limits file that contains the out of limits records for each PTR and a science file that contains the last 3 days of science data. Also archived is a history of commands radiated to the spacecraft during the mission.

3.5 Display of Telemetry

Displaying telemetry in realtime and from the history files is handled by both alphanumeric and graphical displays using either DEC VT340 or Tektronix TK4211 terminals.

A more detailed description of the software architecture is given in the Ulysses SCMS Architectural Design Document (Ref. 3).

4. PROBLEMS AND SOLUTIONS

Following the delivery of SCMS, about one year before launch, it became apparent that the converted system would require upgrading to support the mission fully.

4.1 Excessive CPU loading

Extensive tests using the flight model spacecraft were performed at the European Space Technology Centre (ESTEC) in Holland in late 1989 and early 1990. This testing revealed that the VAX CPU was 100% utilised during periods when both SCMS and flight dynamics software were running simultaneously. This was especially the case when graphics were being used by SCMS and flight dynamics.

The solution to this was two fold. Firstly, an additional processor was added to the system and secondly, a re-write of the graphics subsystem was performed on SCMS. This re-write replaced the all purpose and highly CPU intensive Graphics Kernel System (GKS) with a tailored system using Tektronix commands to drive Tektronix TK4211 graphics terminals. This also

took the graphics facility from one graph with four parameters available per screen to up to twelve parameters and a maximum of eight graphs per screen with a choice of fifteen colours, eight line styles and marker types. The interface for creating graphs was also improved to use menus and contained useful facilities for creating new graphics proformas.

4.2 Long Term Data Retrieval

The converted SCMS uses History Files that contain only the last 15 days of data with older data being lost. This is obviously not acceptable if one is to carry out long term trend analysis or study of an anomalous spacecraft event beyond 15 days when data on the Short History Files would be lost.

The solution to this problem was again two fold: first was the development of the Long History File system which is basically the same as the normal telemetry system but with a history file that contains one format of telemetry every 24 minutes and is sized to allow enough records to cover the duration of the prime mission. (Instead of every 32 seconds with space for fifteen days in the short history file.)

In this way development time and cost were kept to a minimum and, as all user interfaces remained the same, operations team members could easily access the data without special training.

The second part of long term data access allows for a resolution of greater than 24 minutes. This uses data read into an adapted version of the SCMS from Master Data Record tapes which are routinely produced by the JPL Data Management Team. This again uses the standard SCMS for data access by the operations team.

Both the graphics and long history file system upgrades provide powerful tools for data analysis. (See Figure 2 - Example Graphical Display)

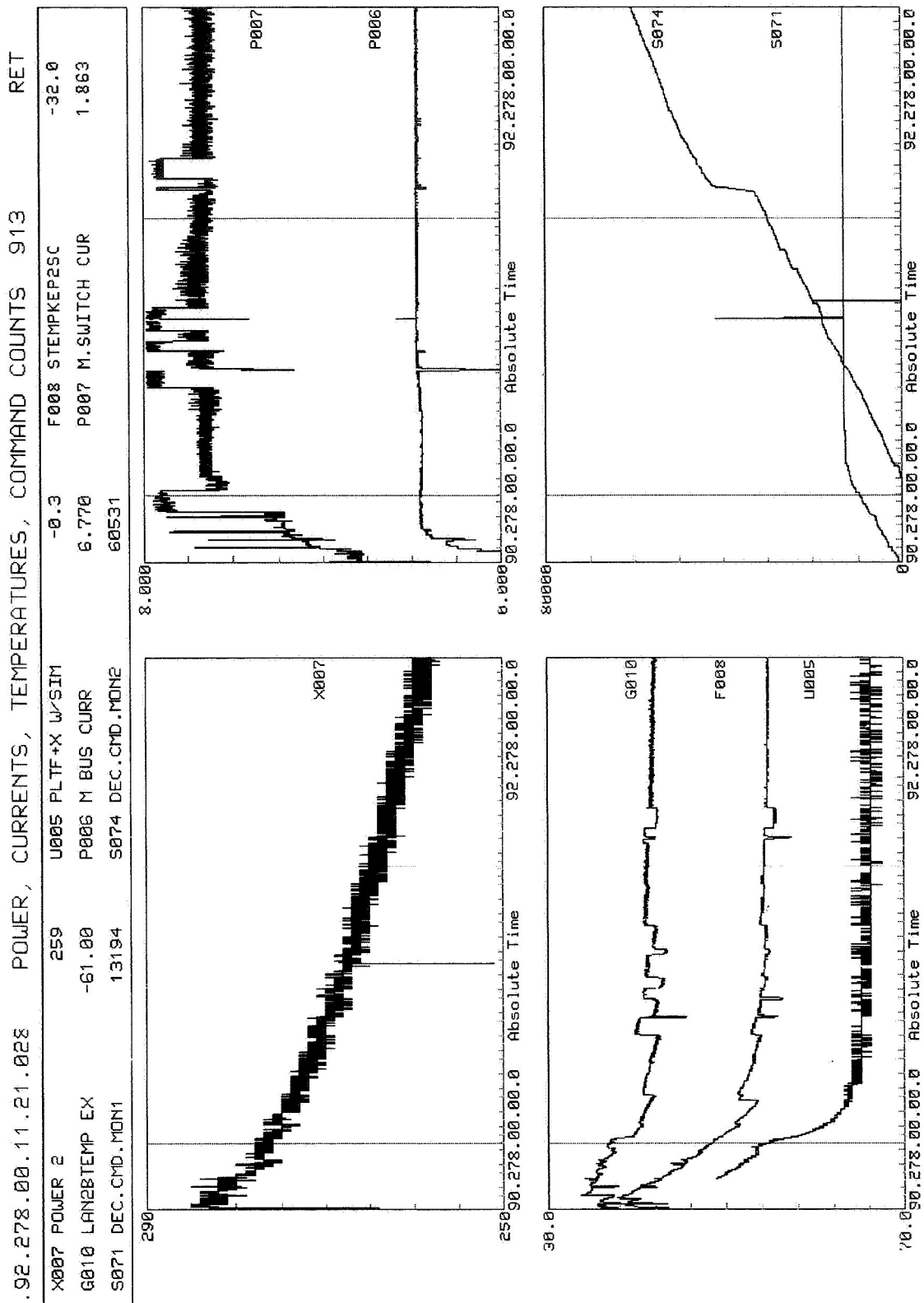


Figure 2 - Example Graphical Display

4.3 Database Editors

As mentioned earlier much of the processing carried out by SCMS involves the use of various databases and following the successful launch of Ulysses in October 1990 improvements were made in utilities to edit these. The original database design was based on the ESA MSSS card system and editing using a standard text editor was error-prone. To reduce the risk of error a menu driven editing system was created for each of the databases. These have been in use since mid 1991.

4.4 Further Post Launch Enhancements

In addition to SCMS enhancements, a need arose for the operations team to have access to information not provided directly by SCMS. These additions were made as functions to be run directly under the VMS operating system. However, the need to understand the operating system was considered an unnecessary addition for the use of the system and so these functions too are accessible via a menu under SCMS. The functions include telemetry statistics packages and a telemetry map function to provide a picture of the whole history file and its contents in terms of data type and quality.

5. FUTURE ENHANCEMENTS

Ulysses has passed Jupiter and is currently leaving the ecliptic with some three years of its prime mission left. As this mission has developed and the operating team refined their needs so SCMS has been enhanced to meet those needs.

The weakest point for SCMS is commanding and as mentioned earlier the MCCC system currently being used for down-link from and up-link to the Ulysses spacecraft is being replaced

as part of a JPL program to modernise the ground system and interface to the DSN. This will bring with it a chance to update the Ulysses command interface for the operations team.

CONCLUSIONS

Following the delay of launch the Spacecraft Control and Monitoring System was successfully transferred from the ModComp to a VAX environment. As no simulator was available for system testing it was less than a year before the scheduled launch that the system's deficiencies became apparent. Additions and modifications were made before the launch with the spacecraft operations and software development team working together to ensure a *user-friendly* system was completed in a timely manner.

Since launch, the SCMS has performed almost flawlessly, although many minor upgrades have been made to enhance the system as a result of new user requirements. With the arrival of AMMOS and as new needs arise these upgrades will continue throughout the mission.

ACKNOWLEDGEMENTS

The authors wish to thank Dr. R. Corkill and Mr P. Pitkin (both of Cray Systems) for helpful comments on a draft version of this paper.

REFERENCES

1. ESA Software Engineering Standards ESA PSS-05-0 Issue 1 (1987)
2. Corkill, Richard 1990. *Ulysses Spacecraft Control and Monitoring Software Paper*.
3. Ulysses SCMS Architectural Design Document Issue 1 Revision 2 (1992)

N94-23863

ULYSSES LOG 1992

Raúl García Pérez
D. Spacecraft Operations Manager

European Space Agency (ESA), European Space Operations Centre
(ESOC), Darmstadt, Germany
Currently working at the Jet Propulsion Laboratory (JPL), Pasadena, USA

ABSTRACT

The Ulysses Log tells the story of some intriguing problems that *we* (=The Spacecraft Team) have encountered. Ulysses was launched on 6 Oct 1990 and made the fastest trip to Jupiter (8 Feb 1992). As it is going out of the ecliptic, let us review some of the Ulysses Log Entries.

Summary:

1. INGENIOUS MANOEUVRES

- 1.1 The Nutation Anomaly
- 1.2 Escape from Direct Sun Pointing

2. TELECOMMUNICATION PROBLEMS

- 2.1 Telecommand sensitivity
- 2.2 Solar Interferences

3. SURPRISES

- 3.1 Spontaneous Reconfigurations
- 3.2 Jupiter. Expect the Unexpected
- 3.3 A Computer Anomaly (CTU2)

Key Words: Anomaly, Nutation, Jupiter, Conjunction, Doppler, Spontaneous Reconfiguration.

1. INGENIOUS MANOEUVRES

1.1 The Nutation Anomaly

The Sun *unevenly heats* a flexible boom as the Spacecraft rotates. This is enough to cause an important dynamic disturbance.

Ulysses has 3 flexible booms:

- *Two Wire-Booms* that rotate in a plane perpendicular to the Spacecraft Spin Axis.

They are located on either side of the Spacecraft Body and their combined length tip to tip is 72.5m (237.86 ft). The centrifugal force of a tip mass keeps them straight.

- *One Axial-Boom* that bends slightly due to the centrifugal force, but remains close to the Spacecraft Spin Axis. Its length tip to root is 8m (26.66 ft).

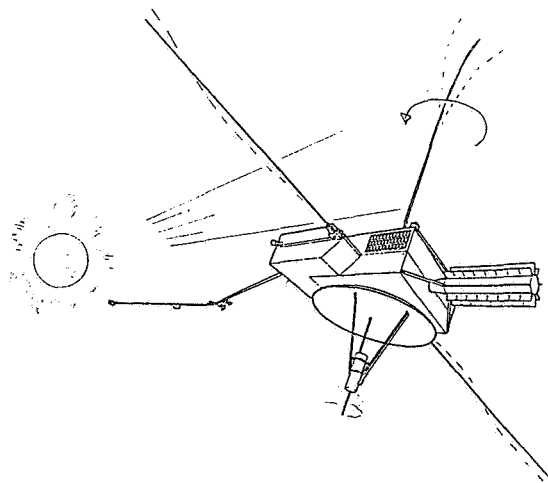


Fig. 1: Ulysses and its flexible elements.

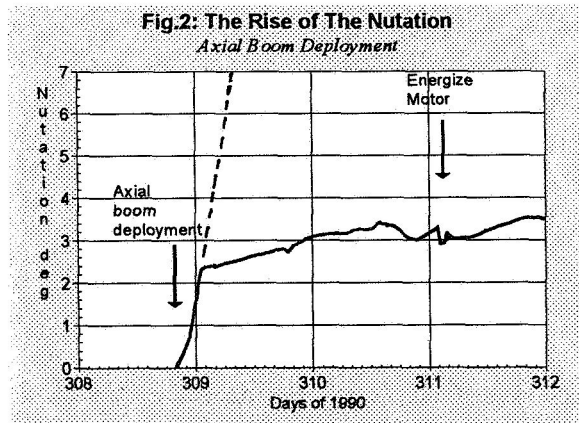
1.1.1 When the Boat Rocks

Deploying a large sail in a strong wind is a delicate operation for a sailing boat. Similarly, deploying the flexible booms in a strong solar wind was a difficult experience for the Ulysses Spacecraft.

The Solar thermal energy caused the axial boom to oscillate. This oscillation excited an undesirable *wobbling* of the Spacecraft body, which is called *Nutation*.

Ulysses Nutation causes the following *effects*:

- **Risk of structural damage.** The axial boom cross section can collapse into a flat tape permitting its storage during the Launch Phase. If the axial boom bends it could wrap around and damage any structure or rip the thermal blankets.
- **Ulysses cannot transmit data to the Earth** because the antenna loses Earth pointing. In the early Mission when Ulysses was closer to the Earth we resorted to the S-Band, which permits wider off-pointing. This will not be possible in the future due to the limited range of the S-Band Transmitter.
- **Degrades The Science Data.** Instrument pointing oscillates and significantly degrades the Science Data.
- **Spacecraft functions run unsynchronized.** The Spin Phase Reference is corrupted. This alters the Spacecraft Manoeuvres and degrades further the Science Data.



The Nutation can be regarded as a small energy imbalance and many normal disturbances would cause it. For this reason any spinning spacecraft would use one of the following two *Methods of Nutation Control*:

- **Passive.** A friction device dissipates the energy of the undesirable motion. Ulysses uses this method.
- **Active.** A well designed manoeuvre precisely corrects the imbalance.

1.1.2 What did we (=The Spacecraft Team) try?

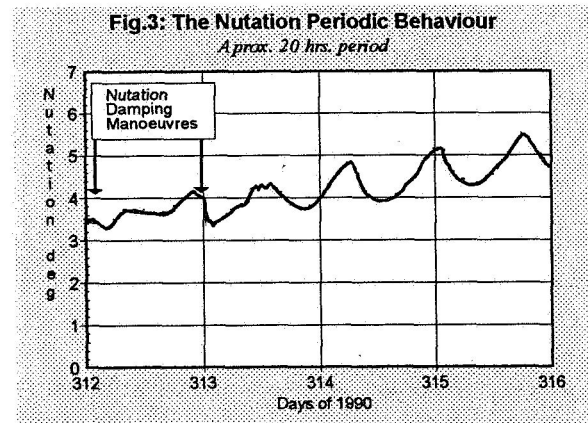
The *Passive Nutation Dampers* were initially flooded by the continuous solar energy input to the system. After 6 long hours of exponential growth The Nutation finally reached an equilibrium at 2.5 degrees full cone (fig. 2).

In the next 4 days the Nutation Amplitude grew up to 4 degrees.

In this early days we tried 2 *Active Strategies*:

- **Energize the boom deployment motor** to eliminate any free-play and stiffen the boom root. The Ulysses flexible booms cannot be undeployed because the motors run in a single direction and stall at the end.
- **Execute Nutation Damping Manoeuvres.**

Both had a very limited effect and the Nutation continued to increase up to 6.2 degrees.



After the Nutation Damping Manoeuvres, the Nutation Amplitude started a *20 hours periodic behaviour* (fig. 3). This was probably a clock-work instability of the equilibrium between the Nutation and the Dampers. As with many other phenomena, this remains unexplained.

The *Resonance Phenomena* were not dominant in the Ulysses Nutation. The analysis of the resonances of the flexible elements indicated that lowering the spin rate could reduce the Nuta-

tion. But from the rigid body theory alone the Nutation would increase. When finally approved the *spin-down manoeuvre to 4.9 rpm* actually increased the Nutation Amplitude.

A corrective *spin-up manoeuvre to 5 rpm* reduced the Nutation below the original level, but the Nutation regained its original amplitude in 24 hours. In further attempts we brought the spin rate up to 5.2 rpm, but the Nutation growth consumed all the initial improvements.

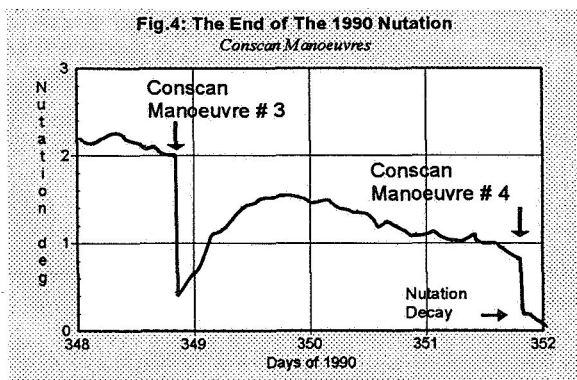
1.1.3 The Solution and the Future of the Nutation

Finally we (=The Spacecraft Team) tried the *Conscan Manoeuvre*. This is an Automatic Earth Pointing Manoeuvre which uses the up-link signal as a beacon to track the Earth.

The Conscan Manoeuvre was designed to have some Nutation damping properties and in fact it is very effective in this respect.

The *Conscan Manoeuvre rapidly reduced the Nutation to 0.25 full cone* (fig, 4), and the success was fully repeatable. To conserve this reduction it is necessary to keep activated the Conscan Manoeuvre, whose fuel consumption is not excessive.

The conditions that produce the Ulysses Nutation will recur in 1994 and 1995. This periods contain the *Solar Polar Passes* which constitute our *Prime Mission*. But we are confident because the Conscan Manoeuvre is an effective tool to prevent the Ulysses Nutation.



1.2 Escape from Direct Sun Pointing

The Ulysses Attitude and Orbit Control System (AOCS) has a *blind spot* in the direction of the Sun. It has the shape of a circle with a radius of 1.25 degrees and centered on the Sun.

Ulysses loses its attitude reference when the Spacecraft main axis is pointing within this circle. The spin-synchronous functions go wrong and this includes the thruster firing for the manoeuvres. In principle Ulysses cannot manoeuvre out of the blind spot.

Nominally we should avoid falling into this trap. But in case of accident, there were no other means of escape than *waiting until the Sun moves away*.

The challenge for us (=The Spacecraft Team) was to *find a quicker alternative* by combining the existing Spacecraft capabilities.

The Conscan Manoeuvre is again part of the solution and the strategy is as follows:

- *Program a clock-work Spin Reference Pulse* to substitute the one that depends on the Sun sensors. The Spacecraft does not know where the Sun is and the phase reference is artificial, but the Spin Period is valid.
- *Activate the Conscan Manoeuvre*. Using the artificial reference The Spacecraft starts tracking the Earth, and the Earth pulls Ulysses out of the blind spot.

In this configuration the Spacecraft gets a totally wrong idea about its inertial attitude. But it is able to manoeuvre precisely towards the Earth. Indirectly, *the Earth provides the missing reference*.

We were close to use this strategy during the second Opposition (Inferior Conjunction). The Sun Avoidance Manoeuvre failed at the edge of the blind spot. But the Sun Reference did not totally fail and we were able to come out with a conventional manoeuvre.

2. TELECOMMUNICATION PROBLEMS

2.1 Telecommand Sensitivity

The acquisition bandwidth for the Telecommand Subcarrier is quite narrow (± 1.7 Hz). This lowers the noise in the Telecommand Decoder but demands a very accurate uplink frequencies. We (Spacecraft Team) have two problem areas:

- *The DSN Command Processing Assembly (CPA) provides valid subcarrier frequencies with sufficient resolution. But their accuracy can degrade beyond the Ulysses requirements. Although the CPAs have a command verification loop to sense the discrepancies, its alarm limits cannot be set narrowly enough.*
- *The Doppler Effect must be compensated. The Spacecraft velocity relative to the Earth causes a frequency change (Doppler Effect). This must be periodically corrected by making the opposite change in the Telecommand Subcarrier Frequency and Bit Rate. The Ulysses injection velocity was the highest ever. Midway to Jupiter the velocity relative to the Earth reached 41Km/sec and created the maximum Doppler Effect.*

2.2 Solar Interferences

During Solar Conjunctions the *Solar Corona* causes radio interference. The phenomenon is well known. But the Spacecraft peculiarities and

the changing solar conditions makes it different every time.

In principle the S-Band uplink should have been more affected than the X-Band downlink. But the downlink was in coherent mode and incorporated also the uplink interference. The coherent mode was necessary for the Solar Corona Sounding Experiment.

The Ulysses X-Band Telemetry was quite affected by the Conjunction while the S-Band telecommands were almost unaffected. Only one command failed during the Conjunction Period.

The *Spectral Broadening* of the signal was the predominant phenomenon. The received frequency varies all the time and the carrier spectrum becomes a bell-shape rather than a line (fig. 5).

To fight the Spectral Broadening we asked DSN to *widen the Receiver and the Subcarrier Loop Bandwidth*. This technique was used by previous projects and it is a trade-off between Acquisition Threshold and Spectral Broadening.

Another observation was that the *measured Signal to Noise Ratio (SNR)* may not represent the real data degradation. The data quality was sometimes very bad while the SNR was above the theoretical threshold. The author attributes this phenomenon to the fact that the Solar Interference can be *non-white or non-gaussian*.

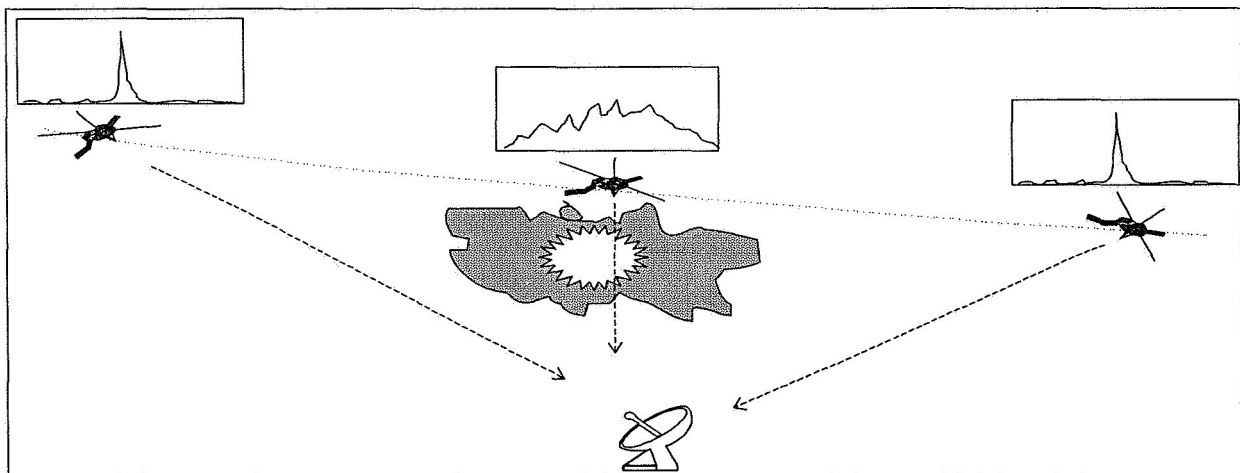


Fig. 5.- Spectral Broadening during Conjunction

3. SURPRISES

3.1 Spontaneous Reconfigurations

Two types of *Spontaneous Reconfiguration* have occurred:

- *TWTA Change-over*: The TWTA (Traveling Wave Tube Amplifier) is the X-Band Transmitter output stage. There are two of them plus a switch-over logic triggered by a power transient on the active tube. The Change-over occurred once, but its cause did not appear in the telemetry.
- *DNEL (Disconnect Non Essential Loads)* is an on-board protection that disconnects all the power consumers not needed for survival.

DNEL may occur for several reasons related to a power system overload. But it might also occur due to an upset in the logic circuits.

DNEL happened during a quiet period of the ecliptic mission. One telemetry format was perfect and the next one was evidencing DNEL. Once again the cause for the Spontaneous Reconfiguration did not appear in the telemetry.

A fast transient may have been missed because the regular sampling rate of the engineering parameters is 1 sample/64 secs. Therefore we will never know whether the Spontaneous Reconfigurations were due to a genuine power transient or not. But their occurrence significantly affected the Jupiter Encounter Planning.

3.2 Jupiter. Expect the Unexpected

All four Spacecraft to encounter Jupiter before Ulysses reported anomalies related to the harsh radiation environment. *Some of the effects on Pioneers and Voyagers* were:

- Spurious Command Execution
- Spontaneous Change of State
- Temporary Instrument Saturation
- Permanent Instrument Degradation

- On-board Computer halted frequently
- Spacecraft Clock Corruption
- Oscillator Frequency Shifts

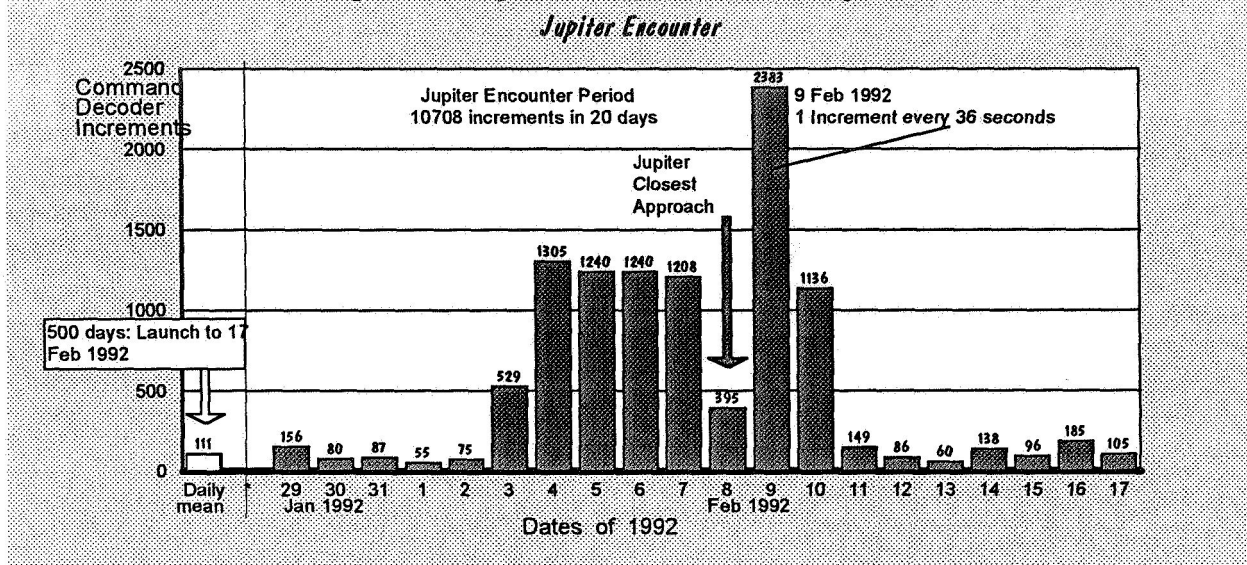
This was an important lesson to learn. Protecting Ulysses against Radiation was a strong requirement during Design and Integration. Even so, *we (=The Spacecraft Team)* prepared procedures for versions of the above that might apply to Ulysses. Many *Contingency Procedures to protect saturated sensors* were applied. Fortunately some other Contingencies like the Spontaneous Reconfigurations or the Spacecraft Clock Reset did not occur during Jupiter Encounter.

To make possible the Operations Timeline became the real problem for two reasons:

- *The high power demands during Jupiter Encounter* could not be easily accommodated. The RTG (Radio-isotope Thermoelectric Generator) was delivering less power than predicted. The Science Teams agreed upon a *power-sharing timeline*. The Spacecraft Team reduced to the bare minimum the non-science power consumption and all the requirements were successfully met.
- *Command Activities were very Complex* because of the following three requirements:
 - » *Command Rate*: this was so high that Instrument's commanding time conflicted. During Jupiter Encounter we sent as many as 20% of all the commands that were sent before (fig. 6).
 - » *Accurate Command Timing*: it was required to within 5 seconds.
 - » *Flexibility*: Many Science Teams wanted to command in near real-time based on their observations.

The intense operations during Jupiter Encounter went like clockwork. *We attribute the success to the highly detailed Timeline, and the good communication with the Science Teams*. The excitement of the Science Teams about the valuable scientific results obtained (Ref. 4) was an excellent pay-off for all the efforts.

Figure 6 : Ulysses Command Activity Peak



3.3 A Computer Anomaly (CTU2)

The Central Terminal Unit number 2 (CTU2) is a redundant computer, that belongs to the Data Handling System (DHS). Its only purpose is to take over in case of a CTU1 failure.

Ulysses (in flight) has used CTU2 twice:

- In the Post-Launch Redundancy Check-out all the CTU2 functions were *normal*.
- In the Post-Jupiter Redundancy Check-out the CTU2 Telemetry Generator showed a *minor anomaly*.

Every 16 telemetry bits, two are linked by a shortcircuit (and-wired, in Electronics jargon). This means that the 2 bits behave as follows:

Real Values	Telemetry Values
0 0	0 0
0 1	0 0
1 0	0 0
1 1	1 1

We are modifying the ground systems to cope with this. If we ever have to use CTU2, the systems will be able to process telemetry with minimum degradation. But CTU1 is in perfect shape and there is no reason to use CTU2.

4. CONCLUSION

We have reviewed in this paper the most interesting Spacecraft Operations Events during the first two years of the Ulysses mission.

The Nominal Mission lasts three years more. The uncharted territory starts really here : Ulysses is now flying away from the Ecliptic Plane towards Solar Latitudes never explored before.

REFERENCES

1. Angold, Nigel, and Hamer, Paul 1992. The Ulysses Spacecraft Monitoring and Control Concepts and Realities. In *Proc. SpaceOps 92*
2. Angold, Nigel. Beech, Peter. García-Pérez, Raúl. McGarry, Andrew. and Standley, Shaun 1992. Ulysses Operations at Jupiter. In *ESA Bulletin No. 72, Nov 1992*
3. Beech, Peter. and Meyer, Donald. Post-Launch Operations . In *ESA Bulletin No. 63 , Aug 1990*.
4. Ulysses at Jupiter. 15 Articles in a Ulysses dedicated issue: *Science vol. 257, 11 Sep 1992*
5. Ulysses Instruments. Dedicated issue: *Astronomy and Astrophysics vol. 92 No.2, Jan 1992*.

GALILEO SPACECRAFT ANOMALY AND SAFING RECOVERY

Ralph R. Basilio and David M. Durham

N94-23864

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California, U. S. A.

ABSTRACT

A high-level anomaly recovery plan which identifies the steps necessary to recover from a spacecraft "Safing" incident was developed for the Galileo spacecraft prior to launch. Since launch, a total of four in-flight anomalies have lead to entry into a system fault protection "Safing" routine which has required the Galileo flight team to refine and execute the recovery plan. These failures have allowed the flight team to develop an efficient recovery process when permanent spacecraft capability degradation is minimal and the cause of the anomaly is quickly diagnosed. With this previous recovery experience and the very focused boundary conditions of a specific potential failure, a Gasptra asteroid recovery plan was designed to be implemented in as quickly as forty hours (desired goal).

This paper documents the work performed above, however, the Galileo project remains challenged to develop a generic detailed recovery plan which can be implemented in a relatively short time to configure the spacecraft to a nominal state prior to future high priority mission objectives.

Key Words: Recovery plan, safing, system fault protection, anomaly

1. INTRODUCTION

The Galileo Project is an investigative mission undertaken by the National Aeronautics and Space Administration (NASA) to acquire, process, and analyze scientific data obtained from the Jovian system.

1.1 The Galileo Spacecraft and Mission

The Galileo spacecraft (Figure 1) was launched on board the Space Shuttle Atlantis and injected into its Venus-Earth-Earth Gravity Assist trajectory by a two-stage Inertial Upper Stage on October 18, 1989. On October 29, 1991 an opportunity of historical significance manifested itself with the first ever flyby of an asteroid (Gasptra) by a man-made spacecraft. Another flyby of a second asteroid (Ida) on August 28,

1993 is currently being planned. Upon approaching Jupiter in 1995, the Orbiter will release a Probe containing six scientific instruments and send it on its way to acquire and relay back scientific data on the Jovian atmosphere. The Orbiter will then be injected into an orbit around Jupiter to perform an intensive twenty-two month scientific investigation of the Jovian system (the planet, its major satellites, and its extensive magneto-sphere). The science investigations will be conducted through the use of four remote sensing and six fields and particles science instruments mounted on the Orbiter.

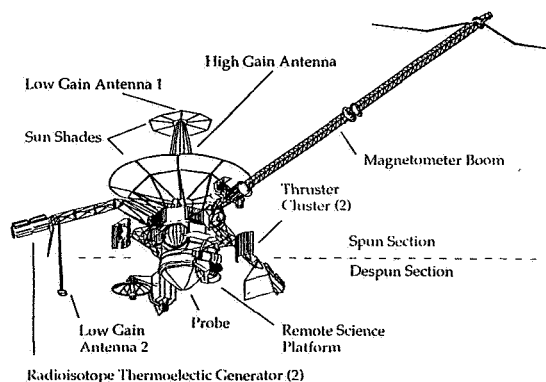


Figure 1 The Galileo Spacecraft

1.2 Sequence Development and Verification

Normally, the spacecraft is controlled and operated through the execution of Nonprivileged Memory Sequence Loads (NMSLs). The duration of each of these sequence loads, used to implement mission activities, may last from a few days to many weeks depending upon mission objectives. Development of the NMSL begins with the activities defined in the Galileo Mission Plan which is used to create a detailed timeline of events. This timeline includes all internal and external resources for execution of the sequence load including timing, ground station allocations, and spacecraft monitoring periods. This resulting timeline is called the Cruise Plan (CP), and takes approximately six weeks to develop. Expansion of the CP is accomplished by specifying the detailed parameters and options of the spacecraft blocks, resulting in a detailed time ordered listing of the commands needed to execute the sequenced activities.

This listing is designated the Profile Design (PD) product and its development typically takes four weeks. The PD product is then converted to the actual NMSL via a sequence translation program and a ground command file is created for uplink to the spacecraft. This phase of the development usually takes another four weeks.

To accommodate unplanned activities or respond to anomalies, real time commands are generated and transmitted to the spacecraft. The duration of the real time command development process can vary from minutes or several days depending on command sequence complexity and urgency.

Validation of the NMSLs are accomplished via manual reviews and software constraint checking of activity plans, command sequence files, and sequence of events listings. Final command sequence verification is primarily performed on the Galileo Test Bed, a facility comprised of flight hardware and software: The prime subsystems comprising the test bed are Command Data Subsystem (CDS) and the Attitude and Articulation Control Subsystem (AACS).

Test bed operations provide the detailed command execution and timing verification which is difficult to ensure just by manual inspection by engineering analysts.

1.3 System Fault Protection Safing Routine

Under some anomalous conditions, faults may place the Orbiter or Probe in jeopardy. To ensure Orbiter and Probe system health and safety, the Orbiter is equipped with autonomous System Fault Protection (SFP). SFP responds with a pre-determined set of commands to safe the spacecraft. The Safing response specifically established for the interplanetary cruise phase is designed to:

- Terminate the active NMSL to prevent unwanted interference with the Safing response.
- Cycle electrical loads off/on to establish a sufficient power margin and be thermally safe.
- Locate the remote science platform to a safe position to protect light sensitive instruments from the sun and thruster plume contaminants.
- Ensure commandability and telemetry according to mission phase dependent requirements.

The Safing response is accomplished in two parts. Safing-1 halts the NMSL and switch off most non-Safing critical switchable electrical loads. Safing-1 places a message in telemetry to alert the flight team of the safing event. Safing-1 ends with a request to initiate Safing-2. Safing-2 establishes a known, safe quiescent state to enable key functions. Safing-2 first halts all AACS tasks except for periodic benign sun acquisitions. Next the telecommunications state is

configured to permit transmission of engineering telemetry. Safing-2 next widens the AACS spin rate and pointing deadbands to more tolerant and robust values. If the safing incident leaves the spacecraft sun pointed, the routine for calling sun acquisitions every 12 hours will continue, thereby causing the spacecraft to track the sun.

Special cruise phase specific instrument safing, propulsion, and thermal safing commands may be issued in real-time to further establish a safe and stable spacecraft state.

1.4 Pre-Launch Safing Recovery Plan

A recovery contingency plan was developed pre-launch. Based on the SFP autonomous responses and entry causes, the plan outlines a process guiding a generalized recovery. The flow of this generalized recovery is as follows:

- Recognize that SFP has been invoked and, if in safing, the exact time the NMSL was aborted.
- Identify and immediately correct safety threatening conditions on the spacecraft.
- Determine exact state of the spacecraft at the time of SFP entry, following the spacecraft's autonomous response, and at the projected sequence restart point.
- Establish cause of the anomaly and correct any persistent fault condition or implement work-arounds for any reduction in spacecraft capabilities.
- Implement spacecraft reconfiguration to the selected sequence restart state.
- Restart aborted sequence or next sequence at its designated load boundary.

However, the plan did not try to be specific, recognizing the following uncertainties:

1. What is the nature of the anomaly causing entry into the safing routine and what corrective action or work arounds are necessary?
2. What was the specific state of the spacecraft at the time of the anomaly?
3. What is the desired state of the spacecraft following recovery and resumption of the nominal operations?

Establishing comprehensive reconfiguration command files or stored sequence restart points which provide opportunities to bring the spacecraft back up to specified states was deemed not practical. A large number of contingency files would have to be generated and significant sequencing memory space would have to be reserved for establishing restart points. The latter presented a memory management difficulty for CDS software analysts. In addition, the cause of an anomaly and fix/workarounds cannot be pre-determined and may negate any pre-generated reconfiguration files and/or restart points in the sequence. Therefore, a predetermined and complete

generic safing recovery contingency plan for any anomaly had been determined not feasible. However, assuming a specific failure, the state of the spacecraft at the time of an anomaly and at a specific restart point, a rapid recovery could be realized.

2. FIRST SAFING INCIDENT

During the final few weeks of the Earth-Venus leg of the mission, the spacecraft was being commanded via a sequence load designated EV-5. An AACCS Star Scanner (SS) calibration, executed during the previous sequence, did not produce all the desired results. Another attempt was then planned for the EV-5 time frame to be executed via real time commands and in parallel with the EV-5 sequence.

2.1 System Fault Protection Safing

Early in the morning on Day Of Year (DOY) 90-015, downlinked AACCS telemetry data indicated that an attitude estimate-related fault had occurred during the previous evening, and that an autonomous abort had brought the subsystem from inertial (Gyro-based attitude estimate) to cruise mode (celestial-based attitude estimate). After much deliberation, the flight team decided to proceed with the transmission of the SS calibration real-time commands even though the assumed spacecraft initial state had changed. It was believed that data gathered during the calibration may give some insight into the fault occurrence. Shortly after the start of the activity, AACCS analysts observed that the clock controller was experiencing difficulty in maintaining control (position and rate) of the Spin Bearing Assembly (SBA). After some time, AACCS SBA-related fault monitors began to trip, eventually leading to an AACCS Power On Reset (POR). This resulted in SFP canceling the EV-5 sequence and requesting spacecraft safing. With the sequence recovery contingency plan in hand, the flight team embarked on a effort that would test the generalized flow defined pre-launch.

2.2 Anomaly Diagnosis

Following verification of all AACCS telemetry measurements, AACCS analysts set off to recreate the anomaly on the test bed. Using cruise mode as the initial state, the analysts observed similar behavior and trips of the same SBA-related fault monitors as observed in flight. Based on these findings and detailed investigative study, a preliminary theory into the cause of the anomaly was developed within hours of the flight incident. The cause of the problem was attributable to a then unknown idiosyncrasy with the AACCS Flight Software (FSW) attempting to maintain a celestial pointing SBA scan type in cruise mode without celestial attitude reference available. Without

celestial attitude reference, the AACCS FSW calculated stator (despun section) attitude based on a noisy spin rate source (i.e. Solar Acquisition Sensor), resulting in SBA control instability. This theory was eventually verified through further test bed simulations and analysis. The spacecraft had been in a different initial state then what the SS calibration had been designed for. Therefore, it was determined that a sequencing error had caused the anomaly, not a failure of one part of the spacecraft system. No degradation of spacecraft capability was noted.

2.3 Recovery and Return to Nominal Operations

To ensure telemetry and thermal safety, real-time commands were sent within hours of the safing incident to reconfigure the telecommunication downlink data rate and propulsion thermal state. Following diagnosis of the anomaly and verifying that no degradation of spacecraft capabilities occurred, the top priority of the flight team was to return the spacecraft to the correct configuration for transmission and execution of the EV-6 sequence, which contained considerably important Venus encounter science activities. The flight team deliberately chose to follow the generalized flow of the recovery plan in a prudent and systematic manner, so as not to jeopardize the Venus science encounter activities.

Although there was a goal to salvage as much of the canceled EV-5 sequence as possible, this was not feasible due to lack of sun-pointed stars needed for reacquiring celestial reference. Since no sun-pointed star sets were available until after the planned final attitude update in EV-5, none of the sequence was salvaged. Consequently, all necessary engineering activities scheduled for execution during the latter portion of the sequence were performed via real-time commands, adding to the flight team's work load. In addition, the first two days of the EV-6 sequence was truncated to allow time for required engineering and science subsystem reconfiguration activities. On DOY 90-037, twenty-three days following the anomaly the shortened EV-6 sequence was successfully transmitted to the spacecraft and executed nominally.

Having taken a prudent and systematic approach, the flight team had an opportunity to verify the generalized recovery process. This gave the flight team confidence in knowing that if future anomalies occur, the flight team would have available a tested contingency plan for reference.

3. SUBSEQUENT SAFING INCIDENTS

As noted earlier the Galileo spacecraft has experienced four entries into Safing to date. The

recent three entries have all been caused by the same condition. This condition was caused by one of the two redundant CDS operating strings detecting a despun CDS POR, causing this string to "shut down" after informing the other string and causing this other string to initiate spacecraft safing. Each of the despun CDS POR events have been spurious transients and not a "hard" failure. The CDS POR indications are believed to be caused by momentary debris induced electrical "shorts" in the SBA interface.

The three CDS POR anomalies have occurred as follows:

- 91-085/13:31:18 SCE CDS B String POR
- 91-123/05:26:51 SCE CDS A String POR
- 91-201/02:09:00 SCE CDS A String POR

3.1 CDS POR Diagnosis and Recovery Planning

A recovery team comprised of representatives from the flight team was assembled. The recovery approach followed the outline given here:

- Ensure the spacecraft state is known, stable, and per predicts , including telecommunication telemetry link performance.
- Provide a preliminary assessment of the anomaly cause and autonomous actions taken.
- Collect more detailed diagnostic data and an analytical assessment of the anomaly.
- Establish project prioritized goals and guidelines for spacecraft recovery.
- Develop an integrated flight team recovery plan.
- Implement the recovery plan systematically and prudently and using standard Project procedures and reviews of each action.

The recovery process was broken into two major activities, recovery of the CDS from its POR state followed by recovery of the spacecraft from its Safing commanded state. Since the anomaly was transient and the SFP response was pre-dominantly within the CDS, the CDS analysts and FSW engineers lead the activity to diagnose the cause of the POR event and define the process for safety restoring both the CDS strings to their pre-anomaly status.

The other major activity was to identify the target date to resume normal sequence activities, identify activities lost during the recovery time frame and reschedule these as needed. Also to be planned were any required spacecraft maintenance activities and spacecraft reconfiguring events to put the spacecraft in the required state for the targeted sequence restart point.

Detailed project prioritized list of recovery goals and guidelines were established as follows:

- Maintain spacecraft health and safety throughout the recovery period and minimize risk of additional faults and operational errors during the recovery process.
- Confirm anomaly fault analysis and determine state of CDS following the anomaly.
- Restore full spacecraft capability to pre-safing state. Restoration of the CDS shall be accomplish at the earliest, practical time using all relevant and establish project procedure and guidelines.
- Establish a spacecraft state to the correct configuration for loading and executing the nominal NMSL at the targeted sequence restart point.
- Develop at the start of the recovery process an integrated recovery plan which shall obtain project approval and shall be used to direct the recovery process.
- Unique, non-standard, and first time spacecraft activities must be simulated on the test bed prior to uplinking to the spacecraft or project must specifically waive the need to do so.
- Recovery activities shall be real-time commanded using immediate executable commands or accomplished by mini-stored sequence of commands - all standard real-time commands or sequence review checks and procedure shall be used to validate all command files.
- The targeted restart sequence shall be simulated relative to the CDS and spacecraft states.

3.2 CDS POR Recovery Activities and Schedules

A full spacecraft recovery plan was devised for the DOY 91-085 anomaly which lasted thirty-one calendar days. The targeted restart point was selected based on the time the project felt was needed to safely implement each of the required recovery activities and to resume the next planned sequence. Stored sequence resumption is important to not burden the flight team with real-time commanding the spacecraft. The plan included the required CDS reconfiguration, spacecraft maintenance activity (normally implemented out of the stored sequence), missed activities from the aborted sequence, and reconfiguration of the spacecraft to the target restart point. The restart point reconfiguration included reconfiguring telecommunications for Engineering High Rate [EHR, 1200 Bits Per Second (BPS)], the High Gain Antenna (HGA) deploy mini-sequence, re-initializing celestial attitude reference, establishing the proper power state of the spacecraft and adjusting the spacecraft attitude for the new sequence. [Unrelated to the CDS anomaly, the HGA deploy sequence failed to deploy the HGA.]

Again, the area of the ecliptic plane in which the spacecraft had been following the sun did not yield many star sets for reacquiring celestial reference, a pre-requisite for returning to nominal spacecraft command sequencing control. This again caused many required spacecraft activities to be performed via real-time commands. Command and control with this method has been found to be taxing to the flight team as a whole.

The CDS POR recovery caused the greatest concern of the recovery process, since it was a never before executed activity on the spacecraft and required cleaning up a number of autonomous actions associated with the CDS software and hardware components. The CDS recovery was broken into four phases:

Phase 1: Error analysis and isolation which provided for detailed telemetry on the CDS state and isolated the down CDS components to allow for reconfiguration.

Phase 2: Reconfiguration and initialization of the down string CDS to establish a state compatible with the processor restart.

Phase 3: Restart the down string processor, synchronize the timing between the two CDS processors and verify proper CDS processing.

Phase 4: Restore and clean-up remaining control flags, telemetry indicators and SFP parameters.

The Project was understandably cautious in implementing each of the phases above, because of the criticality and complexity of the CDS recovery. This part of the recovery alone was spread over thirteen days to allow for complete verification of each phase before committing to the next phase.

The second CDS POR anomaly, DOY 91-123, was similar to the first except that the prime CDS string detected the CDS POR indication. Since normally the engineering telemetry is processed and telemetered through the telecommunications subsystem over the prime string, telemetry was lost as the prime CDS string shut down and the back-up CDS string ran the safing algorithms. This was a previously recognized problem with the SFP in that it does not reconfigure the CDS or the telecommunications interface to allow the back-up CDS telemetry to be downlinked. The initial reaction on the ground following the anomaly was that a ground station problem had prevented proper lock up of the telemetry signal. However, after a review of the SFP and CDS codes it was determined that the anomaly was again a CDS POR but this time on the prime string. Real-time commands were soon processed and transmitted to select the back-up CDS processor for telemetry downlink.

The second CDS POR anomaly recovery followed the same guidelines and approach as the first. The CDS recovery time for this anomaly was reduced from thirteen days to seven days. This was accomplished, because less time was included in the schedule between CDS recovery phases. This was permitted by the Project management once it was determined the anomaly was the same as previously experienced and the recovery process was analogous.

The third and, to date, final CDS POR anomaly occurred on DOY 91-201. Again it was the prime CDS string to detect the POR indication. The one unique aspect of this incident was that telecommunications performance at this phase of the mission could not support the EHR data rate which the previous recoveries were implemented at. Therefore, the recovery command files had to be generated for 40 BPS, the highest supportable telemetry rate at that time. All other events followed closely to the previous recoveries. Again with more confidence in the flight team's recovery action the recovery time was reduced to four working days. This was done by configuring and executing on the same day CDS recovery phases for reconfiguring, restarting and cleaning up of the CDS processors.

It should be noted that during the period of time that the second and third CDS POR anomalies occurred, the Galileo Project was working hard to diagnose and correct the HGA non-deployment anomaly. To maximize flexibility of the flight team and to preclude interruption of a stored sequence during this period, the spacecraft was not executing a NMSL. Therefore the spacecraft recovery following the CDS POR was to essentially reconfigure to a back-ground cruise state.

4. AACs Recovery Issues

To significantly reduce the impact caused by the unavailability of star sets for future celestial reference acquisitions following an anomaly, the flight team investigated the possibility of altering the SFP response to a POR. Instead of unconditionally commanding a sun-point from spacecraft safing every 12 hours and possibly losing celestial reference for some considerable time period, a check or conditional would first be performed. If celestial reference were available at the time of the anomaly and AACs indicated an "earth acquired" status, then the spacecraft attitude would remain unchanged (no unconditional sun-point). In a purely technical sense, earth acquired means that the spacecraft HGA is pointed directly towards the earth within some given error tolerance. With the wide range of commanded spacecraft attitudes planned for HGA anomaly resolution purposes (cool turns and warm turns) and the risk of future CDS Despun POR indications, the flight team's challenge was how to convince the spacecraft that it was always earth-pointed, regardless of the actual spacecraft attitude. AACs analysts determined that all that was required for the "earth acquired" status to be set was to have celestial reference available and to have the AACs FSW track an earth vector. Data contained in the earth vector slot did not have to be related to spacecraft-relative earth motion (i.e. could represent the motion of the sun). All that the AACs FSW was concerned about

was that it was directed to follow motion parameters with the label "earth vector". This change in operating strategy has since been implemented.

5. CRITICAL ACTIVITY PROTECTION

The previous S/C safing events taught us that a quick recovery can be realized if three things are known or planned before the anomaly occurs:

1. The probable cause and autonomous response of the spacecraft are known.
2. The recovery actions correcting the cause and establishing a normal spacecraft state are known beforehand and command files are pre-generated.
3. A specific planned sequence restart point is targeted and the state of the spacecraft at this point is established beforehand

6. GASPRA ENCOUNTER CONTINGENCY

Following the three CDS POR events the Galileo Project had an opportunity to take the events and plan a contingency recovery plan to protect the mission unique Gaspra Encounter. Given the SBA debris theory causing the CDS POR it was considered a real threat to spuriously re-occur prior to the Gaspra encounter scheduled for DOY 91-301. Since the SFP autonomous response had been well characterized it was possible to pre-generate the required command files to correct and restart the CDS state. Looking ahead at the Gaspra sequence designated EE-3', it was possible to determine the exact state the spacecraft had to be in at the start of EE-3'. Thus, a detailed contingency recovery plan could be made such that the duration of the recovery could be minimized. Figure 2 shows a recovery timeline based on the time relative to the occurrence of the POR. Including pad this timeline could be accomplished in as quickly as forty hours for full CDS and S/C recovery, quite a reduction from the twenty-three and thirty-one day recoveries of the first two anomalies. It must be noted without question that the forty hour implementation time was considered only a desired goal (best case). The Galileo project office reserved the right to dictate how fast the flight team should actually implement the spacecraft recovery task. The recovery timeline was to be implemented in the following fashion:

- Identification and confirmation of the anomaly.
- Perform CDS POR recovery and restart.
- Regeneration, if required, of the spacecraft reconfiguration command files based on the actual state of the spacecraft at the time the anomaly occurred.
- Reconfigure the engineering subsystems to the Gaspra sequence initial states.
- Turn on and configure the instruments for the Gaspra science activities.

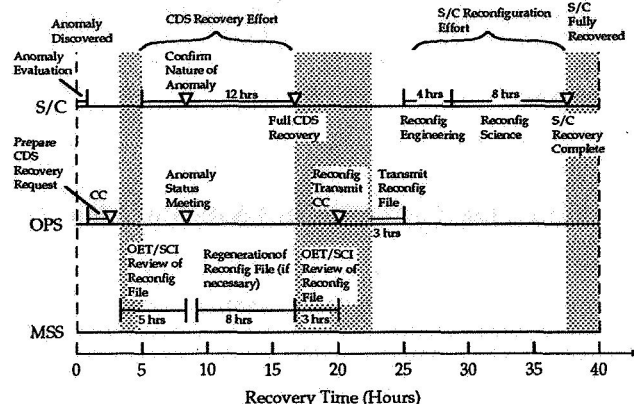


Figure 2 Gaspra Recovery Timeline

7. LESSONS LEARNED

Obviously recovery time from any anomaly is dependent on the cause of the anomaly and any degradation of the spacecraft capabilities following resolution of the anomaly. From the four cases of safing the Galileo spacecraft has experienced, the duration of fault identification to recovery of the fault was reduced from greater than four weeks to inside a week. Experience has shown that familiarity with the cause of the anomaly and its required recovery plan can significantly reduce recovery time. In addition, establishing a recovery approach following prioritized goals and guidelines lead to expedient, yet safe recovery.

8. ACKNOWLEDGMENTS

We would like to thank our colleagues who were previously and are currently involved with Galileo in-flight operations. Also, we would like to acknowledge the encouragement and direction the recovery planning team received from the Galileo Project Office which made it possible to develop and commit to an aggressive and rapid recovery plan to have in place for the Gaspra encounter.

This work was performed at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

N94-23865

MAGELLAN SPACECRAFT AND MEMORY STATE TRACKING; LESSONS LEARNED, FUTURE THOUGHTS

Allen W. Bucher

Martin Marietta Astronautics Group, Denver, Colorado 80201, U.S.A.

ABSTRACT

Numerous studies have been dedicated to improving the two main elements of Spacecraft Mission Operations: Command and Telemetry. As a result, not much attention has been given to other tasks that can become tedious, repetitive, and error prone. One such task is Spacecraft and Memory State Tracking, the process by which the status of critical spacecraft components, parameters, and the contents of on-board memory are managed on the ground to maintain knowledge of spacecraft and memory states for future testing, anomaly investigation, and on-board memory reconstruction.

The task of Spacecraft and Memory State Tracking has traditionally been a manual task allocated to Mission Operations Procedures. During nominal Mission Operations this job is tedious and error prone. Because the task is not complex and can be accomplished manually, the worth of a sophisticated software tool is often questioned. However, in the event of an anomaly which alters spacecraft components autonomously or a memory anomaly such as a corrupt memory or flight software error, an accurate ground image that can be reconstructed quickly is a priceless commodity. This study explores the process of Spacecraft and Memory State Tracking used by the Magellan Spacecraft Team, highlighting its strengths as well as identifying lessons learned during the primary and extended missions, two memory anomalies, and other hardships encountered due to incomplete knowledge of spacecraft states. Ideas for future state tracking tools that require minimal user interaction and are integrated into the Ground Data System will also be discussed.

Keywords: mission operations, state tracking, memory tracking, Magellan.

1. THE MAGELLAN SPACECRAFT

The Magellan Spacecraft is a National Aeronautics and Space Administration's (NASA) planetary mission designed to obtain high-resolution Synthetic Aperture Radar (SAR) images of Venus. Magellan was launched aboard the Space Shuttle Atlantis on May 4, 1989 and arrived at Venus on August 10, 1990. Magellan began SAR imaging the Venusian surface on September 15, 1990 and to date has produced high-resolution imagery of over 99% of the planet's surface.

The Magellan Spacecraft was designed using many spare parts primarily from the Voyager and Galileo spacecraft. The basic design of the craft includes a gyroscope/three-axis stabilized Attitude Control subsystem, a Ni-Cd battery and solar array Power subsystem, a mono-propellant Propulsion subsystem, an S and X Band Telecommunications subsystem, a SAR and Altimeter, and on-board electronics and computers for controlling the craft and peripherals.

The spacecraft contains two main subsystems for controlling its operations, the Command and Data Subsystem (CDS) and the Attitude and Articulation Control Subsystem (AACS). The following paragraphs contain a brief description of the CDS and AACS subsystems. These descriptions provide the reader with information regarding the on-board computer systems integral to the subsequent state tracking discussions.

1.1. Command and Data Subsystem

The CDS computer system is the distributed data system of the spacecraft. This system controls the operations of Magellan by providing a central interface to all components, Figure 1. The tasks performed by the CDS include interpretation of uplink messages, execution of all spacecraft command sequences, fault protection monitoring, and collection and formatting of engineering and science data for transmission to Earth.

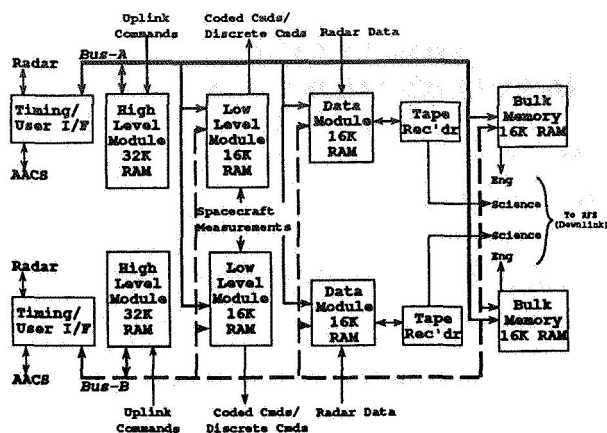


Figure 1 CDS Functional Block Diagram

The CDS computer system is composed of two computer strings which operate in parallel. Only one string is required to be operational for Magellan to function. Thus, the strings are redundant to provide fault tolerance. Each CDS computer system contains two 8 Bit RCA 1802 microprocessors and 80 kilobytes (80K bytes) of active Random Access Memory (RAM) and 80K bytes of redundant RAM for a total of 160K bytes. Approximately 48K bytes of active RAM is used for flight software executable code and parameters. (The CDS flight software contains approximately 600 commandable parameters for software performance customization.) The remaining 32K bytes are used for Command Sequence/Data Storage. The redundant RAM is not normally used to supplement active RAM and the CDS contains no Read Only Memory (ROM).

1.2. Attitude and Articulation Control Subsystem

The AACS is a computer controlled subsystem that performs the tasks required to maintain Attitude and Articulation Control. Tasks performed by the AACS include desired reference generation, attitude determination, momentum management, sequence and scheduling, and fault protection.

The AACS is an assembly of electronic components as shown in Figure 2. The AACS subsystem is a block redundant system with cross-strapping capability between the major components including the sensors, reaction wheels, solar array drive assemblies, and propulsion devices to provide fault tolerance. Each AACS computer contains one 16 bit Applied Technology Advanced Computer (ATAC) microprocessor with 32K words (64K bytes) of RAM. All 32K words of RAM are used for flight software executable code and parameters. The

AACS flight software contains approximately 3000 commandable parameters for software performance customization. Each memory also contains 1K of Read Only Memory (ROM).

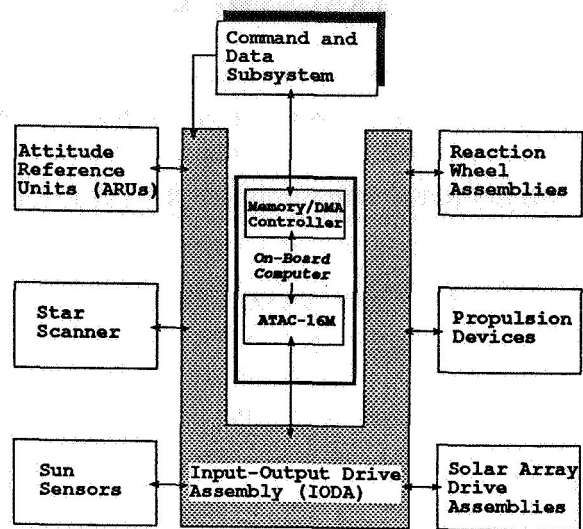


Figure 2 AACS Functional Block Diagram

2. STATE TRACKING OVERVIEW

The two types of state tracking being emphasized in this study are the tracking of critical spacecraft hardware or software states (*Spacecraft State Tracking*) and the tracking of on-board computer memory states (*Memory State Tracking*).

2.1. Spacecraft State Tracking Overview

Spacecraft State Tracking is typically thought of as tracking critical spacecraft components to ensure that the configuration of the spacecraft is known at all times. The spacecraft states that are of interest are those states that may not be readily available via spacecraft telemetry, states that are critical to long term sequence and mission planning, and states that are utilized by the spacecraft fault protection system. For example, the different cross-strapping configurations of AACS hardware are not always directly derivable from spacecraft telemetry due to the usage of switchable primary and secondary hardware relays. These configurations must be derived from the last known state and the state altering commands issued either from the ground or by spacecraft fault protection. The status of critical components must be carefully tracked to assure compatibility with the current sequence of spacecraft activity and the current configuration of the spacecraft fault protection system.

During the life of a given mission, the spacecraft is likely to encounter a variety of mission phases requiring the spacecraft configuration to be altered to meet the goals of that phase. The sequence of events for each phase is planned weeks or months prior to actual implementation on the spacecraft. Mission planners need to plan and create a sequence of events for the spacecraft to perform in the future. Knowing the configuration of the spacecraft in advance presents two unique problems; 1) the current mission phase may have the spacecraft configured differently than future mission phases require, and 2) hardware and/or software failure resulting in fault protection intervention may alter the state of the spacecraft prior to the implementation of future sequences. Knowing the current spacecraft state and the ability to determine the differences between the current state and the state required for future sequences is the goal of effective spacecraft state tracking.

2.2. Memory State Tracking Overview

Memory State Tracking is the process by which the on-board computer memory contents are managed on the ground to maintain an accurate representation of the content of on-board computer memory. Memory is usually partitioned into segments representing flight software code, flight software parameters, and command sequence storage.

The objective of effective memory state tracking is to maintain knowledge of the content of on-board computer memory at any given time. The tracking methods need to supply an effective way to track and report changes, archive historical memory states, and provide interfaces to spacecraft commanding and test facilities. In contrast to spacecraft state tracking, memory state tracking deals only with the content of the on-board computer memory.

3. MAGELLAN STATE TRACKING

3.1. Magellan Spacecraft State Tracking

Spacecraft State Tracking for Magellan is primarily a manual process. Many sequence constraints and spacecraft state configurations are documented in the Spacecraft Block Dictionary (Ref. 1). The Block Dictionary contains the requirements for all spacecraft sequence "blocks". A sequence block is designed to "define the necessary block sequences of commands that will result in execution of spacecraft subsystem events to satisfy required spacecraft operations" (Ref. 1). The group of commands required to desaturate the spacecraft reaction wheels is an example of a sequence block. The Block Dic-

tionary also contains a reference state table used for determining block initial and final conditions including fault protection system configurations. Although the Block Dictionary contains sequence requirements and a reference state table, the implementation of the required spacecraft state configuration is accomplished manually by the spacecraft team. The current spacecraft state is manually determined from telemetry, the state for future sequences is determined manually, and the system fault protection is manually reviewed to insure compatibility with the desired spacecraft state. These manual processes rely heavily on human expertise and discipline to verify all states and configurations. This process worked well while the spacecraft team was at the maximum staffing levels and while the "experts" remained part of the team. Hardships are encountered however, as the spacecraft team size continues to decrease and the experts leave the team.

Another difficult task during major mission phase changes is the preparation of future states for the System Verification Laboratory (SVL). SVL is a high fidelity software and hardware simulator of the Magellan spacecraft designed for flight software testing, sequence verification, and anomaly resolution testing. The SVL has to be cognizant of the actual spacecraft state changes to insure accurate spacecraft state representation and must be properly configured with future spacecraft states to accommodate future sequence testing. When the spacecraft team does not perform all required state changes, the SVL is able to determine the majority of the sequence versus spacecraft configuration mismatches. The SVL is not able to catch all inconsistent states because not all spacecraft components are modeled. With most of the emphasis focused on creating and tracking valid states in SVL to insure valid sequence testing, the SVL has become the default implementation of a spacecraft state tracking and propagation tool for Magellan. Although this method works, the SVL still has to manually implement changes to maintain compatibility with the current spacecraft state. Also, there is not an easy method for the spacecraft team analysts to retrieve information from SVL regarding past or future spacecraft states.

All critical component state tracking was also accomplished by engineers from each subsystem manually recording their individual subsystem states in a spacecraft state table. However, soon after launch, the spacecraft team developed the ability to autonomously track all spacecraft states that are telemetered. The spacecraft team created a Unix script entitled "Moose Query" which queries all

Magellan telemetry from the central database and produces a summary report of all states that are derivable from telemetry. Although this report is still manually compared to a table of expected states, it greatly simplifies collection and recording of the approximately 1000 spacecraft states. In addition to the Moose Query script, the spacecraft team also uses the Data Monitor and Display capability to dump the latest available data (LAD) buffer for spacecraft state tracking. This LAD dump capability is a useful analysis tool when it is not necessary to determine the exact time the state changed but to have a snapshot of the spacecraft states at a specified time. The Moose Query and LAD dump reports are archived to provide a permanent record of the spacecraft state.

All state changes that are accomplished as part of the nominal mission are carefully planned and reflected in the sequence development cycles. Many hardships result from last minute changes to sequence plans due to fine tuning of mission requirements or from unanticipated spacecraft configuration changes. For example, the spacecraft sequence designed to perform reconditioning of the Magellan batteries had been planned and constructed assuming use of transmitter A. However, due to degrading performance of transmitter A, the spacecraft team was forced to switch to transmitter B prior to activation of the sequence. This switch was not reflected in the nominal sequence planning and generation process. The sequence was radiated to the spacecraft and when execution commenced, the spacecraft fault protection, enabled by the sequence, detected that transmitter A was not producing an output signal (because it had been previously commanded off). The fault protection software, assuming failure of transmitter A, issued commands to autonomously switch to transmitter B. The fault protection activation resulted in several telemetry alarms and some short-lived excitement. After analysis it was determined that the commands issued by the fault protection software had no effect because the transmitter A to B swap had already been accomplished by ground command.

This is one example where the spacecraft team was forced to react in an emergency mode because the sequence did not match the current state configuration of the spacecraft. Fortunately the result of the configuration mismatch was benign. The incident did however raise some questions regarding the spacecraft team's ability to manually guarantee that the configuration of the spacecraft is consistent with the current spacecraft sequence and fault protection configuration.

3.2. Magellan Memory State Tracking

Memory State Tracking for Magellan is accomplished by a suite of software tools designed to perform the tasks of flight software loading, flight software parameter tracking, and memory compare and analysis. The software tools are broken into three computer programs designed to carry out the required tasks. These programs are defined as:

1) *Flight Software Loading* - **VMFLOAD** is designed to interface flight software loads from either the flight software development environment, the spacecraft test facilities, or the mission operations environment to the command system. VMFLOAD accepts hexadecimal flight load images and reformats them into uplink messages for processing by the sequence translation software and subsequent radiation to the spacecraft. These "uplinkable" images are also compatible with the SVL to allow testing prior to actual command radiation.

2) *Flight Software Parameter Tracking* - **VMPTRK** is designed to track commanded changes to flight software parameters. VMPTRK reads the Spacecraft Event File (SEF) and extracts commanded changes to tracked parameters and records the new parameter value, the time of the change, and the identification of the SEF that altered the parameter in the VMPTRK Tracked Parameters Database. The Tracked Parameters Database contains a historical archive of all parameter values and the information regarding each update.

3) *Memory Compare and Analysis* - **MEMANAL** is a program designed to read a file representing a computer memory image and file of spacecraft Memory ReadOut (MRO) and identify any mismatches between the ground memory image and the MRO data. Spacecraft MRO data is generated by issuing a spacecraft command designed specifically to "Read Out" a selected area of memory and place the result in the spacecraft telemetry stream. MEMANAL has the ability to compare and analyze any selected portion from either computer system.

After the Magellan flight software and associated parameters were loaded and the spacecraft was configured for launch, all on-board computer memories were read-out and verified against the final flight software builds using MEMANAL. The results of these verifications were baselined as the first ground memory images. The contents of the first VMPTRK Tracked Parameter Database was also verified. These initial configurations were important as all future configurations are derived as deltas to the initial baselines.

The process of memory state tracking was accomplished by tracking commanded changes to parameters via VMPTRK, tracking changes to flight software loads via VMFLOAD, and verifying changes via MEMANAL. Periodically the on-board computer memories are "read-out". The resultant MRO is compared to the latest ground memory image(s) and any miscompares are verified. Once the new MRO is verified, it is archived and placed under configuration management as the current state of the on-board memory. This method provides for periodic recording and validation of the memory state. As an independent check the tracked parameter database is also compared to the MRO. This accomplishes two objectives: It verifies that the content of the on-board parameters is as expected, and it verifies that the content of the tracked parameter database is correct and can be trusted as an accurate recording of critical flight software parameters.

To provide examples of the use of these tools, resolution of two memory anomalies on the Magellan spacecraft will be discussed. The following paragraphs give a brief description of each memory anomaly and the actions taken by the Magellan spacecraft team to resume normal spacecraft operations.

3.2.1. CDS Memory Read Parity Error

On December 31, 1989 the CDS processor detected a memory read parity error indicating that the parity reading from the memory was different from what the processor expected. This is considered a privileged error by the CDS resulting in the spacecraft placing itself in a safe state. To recover from the safe state, the spacecraft team decided to utilize the redundant RAM associated with the suspect CDS processor. The redundant RAM however, was still configured with the launch flight software and parameters and would have to be reconfigured to the current memory state before it could be used. The first step was to read-out the primary RAM to determine the up-to-date configuration and read-out the redundant RAM to determine its exact content. The two memory read-out images were compared, using MEMANAL, to determine the memory image configuration differences. The differences determined the changes required to restore the state configuration of the redundant RAM. Once the required changes were determined, commands were built to update each individual parameter that was out-of-date. Each command was compared with the configuration managed memory image and the VMPTRK tracked parameter database to insure correctness. After the commands were verified, they were tested in the SVL and then radiated to the

spacecraft. After successful radiation the memory image was again read-out and verified, using MEMANAL, before the processor was returned to operation using redundant RAM.

3.2.2. AACS Memory Address Failure

Following Venus Orbit Insertion the off-line AACS computer experienced an intermittent memory address bit failure resulting in an area of off-line RAM that was not accessible by the CPU. Realizing that this configuration was not desirable, the spacecraft team configured the AACS on-board computers to indicate that the off-line memory and CPU were not usable while they established the probable failure mode and solution. Also as a precautionary measure, the entire off-line memory was "back-filled" with a alternating pattern of all 0000's followed by all FFFF's at 16 word boundaries. The 0000's represent no-ops and FFFF's represent traps to the CPU, both of which were benign should the intermittent failure occur again.

The back-fill was accomplished by creating a memory image with the desired 0000/FFFF pattern. VMFLOAD was used to generate a series of uplink messages. The uplink messages were processed by the sequence translation software, tested in the SVL, and then radiated to the spacecraft to neutralize the memory while trouble-shooting could continue.

Spacecraft experts were able to determine a probable failure scenario and also determined that if the problem diagnosis was correct, the problem would eventually "heal" itself (Ref. 2). With the off-line memory loaded with the 0000/FFFF pattern, the AACS was essentially operating in a single string mode. Therefore, the project was anxious to get an operational load into the off-line memory to restore the AACS to redundant dual string operation.

The first step in the process was construction of a valid memory image. The memory image was constructed by utilizing the last configuration managed on-line image and overlaying it with the up-to-date parameters from the VMPTRK tracked parameter database. VMFLOAD was used to generate a series of uplink formatted messages from the memory image. These uplink formatted messages were processed by the sequence translation software to generate the actual uplink files. These uplink files were tested in the SVL and then radiated to the spacecraft. Following successful uplink radiation, the memory image was read-out and verified via MEMANAL. After successful verification the off-line memory and CPU were returned to normal operation.

4. SUMMARY

Magellan State Tracking is primarily a manual function allocated to the sequence design process and the individual subsystem analysts. High level groundwork for the state tracking process was initiated by the sequence block dictionary. This implementation did not, however, facilitate the continuation or automation of the state tracking process required for mission operations. The sequence generation process provides a mechanism to track component states via the initial conditions/final conditions scheme. The states reflected by this scheme also require manual updates whenever state changes occurred in a non-sequence fashion. Each sequence assumes a certain configuration and the spacecraft team did not have an efficient way to implement changes in the latter part of the sequence cycle. The burden of late configuration changes was always borne by the Upload Preparation Group who worked tediously implementing manual edits to the final sequences reacting to late or unanticipated configuration changes.

The level of manual activity required for state tracking was not a concern when the spacecraft team was at maximum staffing levels and contained experts dedicated to analyzing major state changes and the fault protection implications. As the team size continues to shrink and critical personnel leave the team, the worth of a more automated state tracking mechanism appears more attractive. However, at this stage of the project the creation of such a tool is not feasible. Given the budget constraints encountered by the mission operations designers and the repetitive nature of the mapping mission this was probably the correct approach. It is difficult to quantify the savings that an automated state tracking tool would have achieved.

The spacecraft team did develop a method to autonomously track after-the-fact spacecraft states. This method can track current states based on spacecraft telemetry but does not project future state configurations for sequence planning or fault protection configuration purposes. This tool is used primarily to record spacecraft states during critical periods or spacecraft anomalies.

The process of memory state tracking including flight software code and parameters has been a tremendous asset to the Magellan program. The ability to track critical parameters and reload partial or entire on-board computer memories with minimal effort has proven to be an integral part of anomaly resolution and recovery. The three tools discussed above, VMFLOAD, VMPTRK, and MEMANAL, all

utilize common input formats and are compatible with the formats used by the sequence translation software used to generate and translate command sequences. The input and output formats are also compatible with the formats used by the SVL and the flight software development processes. This format commonality has allowed the tools to provide the flexibility needed for mission operations.

5. FUTURE THOUGHTS

Spacecraft and Memory State Tracking as ground analysis tasks are an integral part of mission operations. Spacecraft designers should strive to place all spacecraft state information in telemetry. Memory contents and states can be tracked based on a known configuration and updates to that configuration. The majority of spacecraft and memory state changes are accomplished by the command or sequence process and can be determined via telemetry. An effective state tracking tool must provide an automated and integrated method to:

- Interface with the command system to determine commanded state changes,
- Interface with the telemetry system to determine autonomous changes to states and provide a check-and-balance mechanism with the intended states from the command system,
- Provide the capability to determine required state changes based on the current state and the state required to support the desired sequence or mission phase,
- Interface spacecraft and memory states with the spacecraft test facilities, and
- Provide effective reporting of states at both the management and analyst levels.

With such a large impact on the sequence design and generation process, spacecraft and memory state tracking should be an integral part of the sequence design and generation tools.

6. REFERENCES

1. Spacecraft Flight System Block Dictionary, VRM-SE-025, Rev: Final, Feb. 1991, pages 1-4.
2. Kasuda, Rick, "ISA 8899 AACS Memory B Bit 4 High Addressing Failure", January 4, 1992.

7. ACKNOWLEDGEMENT

The work described in this paper was accomplished by Martin Marietta under contract to the Jet Propulsion Laboratory, California Institute of Technology and sponsored by the National Aeronautics and Space Administration.

SESSION D

MISSION AND SCIENCE PLANNING AND SEQUENCING

CO-CHAIRPERSONS

C. P. Dent, NASA Goddard Space Flight Center, USA
A. M. Tavormina, JPL, California Institute of Technology, USA

Summary	217
A. M. Tavormina	
D.1 A Review of Mission Planning Systems	219
M. Jones, E. M. Sorensen, and T. Wolff, ESA European Space Operations Centre, Darmstadt, Germany; C. R. Haddow, Logica Space and Communications Ltd., London, UK	
D.2 The Mars Mapper Science and Mission Planning Tool	225
M. W. Lo, JPL, California Institute of Technology, Pasadena, California, USA	
D.3 Mission Planning for an Earth Observation Low Earth Orbiter: ERS-1	231
P. Lockyer, Cray Systems, Bristol, UK	
D.4 Integrating Payload Design, Planning and Control in the Dutch Utilisation Centre	237
T. J. Grant, BSO/Aerospace & Systems b.v., Nieuwegein, The Netherlands	
D.5 Voyager Uplink Planning in the Interstellar Mission Era	243
S. H. Linick and K. R. Weld, JPL, California Institute of Technology, Pasadena, California, USA	
D.6 Science Planning and Sequencing for Cassini	249
R. R. Wessen and D. F. Finnerty, JPL, California Institute of Technology, Pasadena, California, USA	
D.7 Techniques and Software for Optimum and Efficient Mission Science Sequence Development	255
D. A. Bliss, JPL, California Institute of Technology, Pasadena, California, USA	
D.8 A Packet-Based Concept for Spacecraft Command Planning	261
V. B. Barnes, Johns Hopkins University Applied Physics Laboratory, Laurel, Maryland, USA	
D.9 Connecting Science and Operations: The Operations Coordinator	267
M. H. Marshall and J. A. Landshof, Johns Hopkins University Applied Physics Laboratory, Laurel, Maryland, USA	
D.10 The "Instant Sequencing" Task: Toward Constraint-Checking a Complex Spacecraft Command Sequence Interactively	273
J. C. Horvath, L. J. Alkalaj, K. M. Schneider, and A. V. Amador, JPL, California Institute of Technology, Pasadena, California, USA; J. N. Spitale, California Institute of Technology, Pasadena, California, USA	



D.11 AGENDA: A Task Organizer and Scheduler	279
I. Fratter, CNES Toulouse Space Centre, France	
D.12 Columbus System Support for Telescience Operations	285
D. W. Lytton, Space Applications Services, Brussels, Belgium; R. Schulze, ESA European Space Research and Technology Centre, Noordwijk, The Netherlands	
D.13 Generic Mission Planning Concepts for Space Astronomy Missions	291
O. T. Guffin and J. F. Onken, NASA Marshall Space Flight Center, Huntsville, Alabama, USA	

SUMMARY — SESSION D

A. M. TAVORMINA
Jet Propulsion Laboratory

Thirteen 20-minute talks were presented in Session D. The audience ranged from 30 to 50 people throughout the day. The speakers were from six countries: Belgium, France, Germany, The Netherlands, the United Kingdom, and the United States. Six entities were represented: the Applied Physics Laboratory at Johns Hopkins University, CNES, the Dutch Utilisation Centre, ESOC, JPL, and NASA Marshall Space Flight Center.

Topics included tools and the processes used to plan space platform activities, balancing spacecraft needs against the science and mission goals. Experiences with at least a dozen spaceflight systems were used as examples by the speakers, including Astro, AXAF, Columbus, Cassini, Eureka,

ERS-1, Galileo, Hubble Space Telescope, Mars Observer, SPOT4, TOPEX/Poseidon, and Voyager.

Several common themes arose during the session that emphasized the importance of the following topics:

- Early understanding of requirements and constraints.
- Building modular, transportable tools.
- Developing very clear team interfaces.
- Building flexibility into the planning and sequence-development process.

N94-23866

A REVIEW OF MISSION PLANNING SYSTEMS

M. Jones, E. M. Sørensen, T. Wolff

European Space Operations Centre,
Robert-Bosch-Str. 5, 6100 Darmstadt, Germany

C. R. Haddow
Logica Space and Communications Ltd., London, UK

ABSTRACT

A general definition of Mission Planning is first given, covering the full scope of an end-to-end mission planning system. Noting the mission-specific nature of most mission planning systems, a classification of autonomous spacecraft missions is made into Observatory, Survey, multi-instrument science and Telecommunications missions. The mission planning approach for one mission in each category is examined critically, the missions chosen being ISO (Infrared Space Observatory), ERS-1 (European Remote Sensing Satellite) and Eureka (European Retrievable Carrier). The paper gives a summary of lessons learned from these missions, suggesting improvements in areas such as requirements analysis, testing, user interfacing, rules and constraints handling.

The paper will also examine commonalities in functions, which could constitute a basis for identification of generic mission planning support tools.

I. DEFINITION AND SCOPE OF MISSION PLANNING

The term "mission planning" is used in this paper to refer to any system which plans the operations of a space system or any of its component parts. A Mission Planning system may plan platform operations, payload operations, on-board data handling (recorder or memory). It may also plan ground operations associated with the foregoing (ground station activities, payload data processing, product dissemination).

The inputs to a planning system from the payload users can be either of the following:

- requests for particular data, products or measurements
- specification of specific payload operations, which may vary from high-level specification of operations required to binary command data at the lowest level

The main outputs will be the *final operations plan*, which is passed to the commanding system at the control centre for translation into uplinkable form and execution. An appropriate set of mission rules and constraints governs the derivation of the operations plan from the user inputs.

II. CLASSIFICATION OF MISSIONS

Four mission types are identified: Survey, Observatory, Multi-Instrument PI Missions and Telecommunications.

Observatory Missions: the spacecraft has one main instrument (usually a telescope, although there may be several focal plane instruments). Users are allocated observing time during which they have dedicated usage of the instrument to study particular objects. Spacecraft and associated centralised ground facilities can be considered to be the "Observatory". Operationally, payload control is typically carried out by a centralised operating agency, but end users are given facilities to control the instrument during their allocated slots. Observers are normally at the ground observatory facilities during their operational periods.

Survey Missions: these carry out repetitive observations on a large scale. The spacecraft will typically have a single payload instrument, or a small number of closely related ones. The spacecraft and payload are normally operated by a centralised agency on behalf of a number of end users, who can request specific observations within the planned strategy.

Multi-Instrument: the spacecraft has a number of (in principle) independent experiments, each provided by a separate "Principal Investigator" (PI). A centralised agency will operate the platform, but PIs are responsible for operation of their experiments, using services provided by operating agency. PIs can be remote from the control centre, submitting requests and getting data from their experiments electronically.

Telecommunications Missions: the spacecraft has a transponder(or transponders) to provide communications between ground stations ("fixed" service) or between another spacecraft and ground(data relay service). The spacecraft and its payload is usually operated by a centralised agency on behalf of the end users. Allocation of transponder communication channels to users (or subscribers) is made as a function of service requests.

The above classification of missions shows the common characteristic that missions are usually operated by a central agency on behalf of the end users. This reflects the fact that there is normally only one uplink route for commands and that operation of the platform and the payload are often interlinked (e.g. through resource dependencies). In all cases some sort of mission planning is needed to process user requests.

III. MISSION PLANNING FOR SURVEY MISSIONS: ERS-1

The first European Remote Sensing Satellite (ERS-1) was launched in July 1991 by ARIANE-3. The main mission of ERS-1 is the monitoring of the state of the ocean and ice. It is in a near polar orbit at an altitude of c. 800 km. Its main payload is the AMI (Active Microwave Instrumentation) which comprises a synthetic aperture radar (SAR), a wind scatterometer and a radar altimeter. An on-board data handling system (IDHS) includes a tape recorder, which records data from all instruments except the SAR; the tape recorder operations have to be

scheduled. The SAR can only be operated in ground station contact, so that the data acquired can be downlinked in real time.

The prime ground station both for TT&C and payload data acquisition is at Salmijaervi near Kiruna, Sweden and the control centre (the ERS Mission Management and Control Centre - MMCC) is at the European Space Operations Centre (ESOC), Darmstadt, Germany. Services for end users are provided by the ERS-1 Earthnet Central Facility (EECF) at ESRIN, Frascati, Italy.

The ERS Mission Planning System (MPS) is responsible for planning (1) spacecraft and payload operations and (2) ground station operations, in particular data acquisition and data processing

ERS-1 operates mainly autonomously based upon uplinked command schedules covering 24 hours. Mission Planning is therefore central to ERS-1 operations.

Mission Planning for ERS-1 is split into two parts

1. Preparation of the Preferred Payload Exploitation (PEP) from external user requests.
2. Preparation of the Detailed Mission Operations Plan (DMOP) from which the command schedule file uplinked to the spacecraft is generated.

Step 1 (the user-oriented part) takes place at the ERS-1 Earthnet Centre Facility, Step 2 takes place at ESOC, the DMOP being produced by the ERS-1 Mission Planning System (MPS). The DMOP is passed to the MMCC for execution. It is also transmitted back to EECF for information. A restituted DMOP, reflecting the actually executed operations, is also sent to EECF. Plans cover 3 days and are available at least 24 hours before execution.

CRITICAL DISCUSSION

The ERS-1 MPS is the most ambitious and expensive mission planning system developed at ESOC. The split of responsibility between ESRIN and ESOC is in principle sound, with ESRIN collecting and filtering user requests and then passing them to the MPS at ESOC in a high level form for conversion into the final schedule for uplink.

The following problems are noted

Rules and Constraints: the rules and constraints for mission planning are specified in a Budgets and Constraints Document. The rules have been hard coded into the MPS. As a consequence it is not easy to make changes in them. Ways of incorporating rules and constraints into an early modifiable database or knowledge base need to be examined.

Interface between control centre and user centre; Definition of this interface was not done under sufficiently rigorous control. This highlights the need for preparing a complete Interface Control Document under configuration control at a fairly early stage. Furthermore, the MMCC/EECF interface does not provide sufficient constraints information to allow the MPS to complete its work without discussion between EECF and ESOC Mission Planning staff.

Testing: testing of such a complex system is difficult and needs considerable user support (i.e. from operations personnel). Provision of user-defined test procedures for use by developers was found to be necessary, as was more formal acceptance tests by operational mission planners

IV. OBSERVATORY MISSIONS: ISO

The Infrared Solar Observatory (ISO) is planned for launch in 1995. It will carry a telescope with four focal plane instruments. It will fly in a near geosynchronous orbit, which gives, on average, 10 hour passes over the prime ground station situated at Villafranca, near Madrid, Spain. ISO operations are carried out in real-time according to a ground-executed schedule, i.e. not an on-board schedule like ERS-1 (sect. 3.1) or EURECA (sect. 3.3). Only one of the four focal plane instruments operates at any given time. Observations can vary in length from 2 minutes to 10 hours. There are no resource constraints.

Mission planning for ISO is split into a user-oriented part (MPP1) under the responsibility of ESA's Space Science Department and an operations oriented part (MPP2) under ESOC responsibility. The mission planning concept involves an iteration between these parts: Firstly, a Planning Skeleton file (PSF) is prepared by MPP2, in which slots are identified for payload operations and for platform operations. Secondly, MPP1 processes observation requests and produces an instrument command schedule to fit in

the allocated slots. The instrument commands are expressed as mnemonics, referring to pre-prepared binary command sequences. MPP1 transmits this PSF to MPP2 as the Planned Observations file (POF). Thirdly, MPP2 inserts flight dynamics (e.g. attitude adjustment) commands necessary for the instrument observations and merges in command requests from ESOC operations staff. The result is a Central Command Schedule (CCS) for submission to the commanding system.

CRITICAL DISCUSSION

A number of very positive points about the ISO approach are:

- Use of the PSF technique, allowing platform and payload operations to be planned separately. This appears to be particularly applicable for missions like ISO.

- A very rigorous approach taken to preparation and control of interfaces (via formal Interface Control Documents) also resulted in early definition of the interfaces and functionality of the ISO Mission Planning System, so development of both parts could proceed independently.

Another interesting feature has been the application of a rigorous separation between platform and payload, the idea being separation of responsibilities for platform and payload control the former being that of the operations control centre (ESOC) and the latter that of the Observatory. This has led to what is in effect two control systems constrained by the necessity of having one uplinker. These two *de facto* control systems are distributed over a complex configuration of computers. This affects both mission planning and the on-line control systems. This has several consequences

- End to end testing of the whole mission planning and commanding process is made more complex, since instrument command execution verification occurs within instrument workstations rather than on the real-time control system.

- The split of operational responsibility has led to a large number of file transfers between systems. This is related to the large amount of hardware associated with the different groups. This may have performance and reliability impacts.

- Instrument commands sequences are prepared by MPP2 in uplink form. Input from MPP1 to MPP2 is in the form of mnemonics referencing these pre-prepared uplink blocks. A command translator converts the mnemonics to the uplink form. This is cumbersome and could be inflexible and error prone if instrument command sequences need to be changed.

3 MULTI-INSTRUMENT MISSIONS: EURECA

EURECA was launched by the Space Shuttle on 29th July 1992 and is planned to be retrieved by the Shuttle about April 1993. EURECA carries a complement of some 15 experiments, serviced in some cases by core (or common) facilities (furnaces etc.). The experiments are in the main independent, except where they make use of core facilities. A number of discipline areas are covered, including microgravity (protein crystallisation and materials science), solar science, in-orbit data relay and spacecraft technology (e.g. a new type of ion thruster). EURECA flies in a 28° inclination circular low-earth orbit at an altitude of c.500km. Two ground stations at Maspalomas (Canary Islands) and Kourou (French Guiana) are being used during the routine phase. Station passes are short (2-10min.) and occur in a consecutive set of 4-5 orbits. The contact ratio with the ground is thus very low and payload operations mostly occur autonomously when the spacecraft is out of contact with any ground station. This makes planning of payload and platform operations essential, especially as the resource constraints on EURECA (primarily power, cooling and data storage) are such that it is not possible to run all experiments in parallel. Thus EURECA mission planning is used to schedule operation of the experiments such that available resource limits are not exceeded. Passes are devoted mainly to uplinking of schedules and dumping of data from on-board memory.

EURECA mission planning can be split into two parts: pre-launch planning and operational planning.

Pre-launch planning; This produces a baseline plan for the entire mission based on certain assumptions, for example the amount of power generated by the solar panels, the efficiency of the on-board cooling loop and the expected orbit and launch time. The baseline plans served to do the following;

1. Determine that sufficient time could be allocated to each experiment to satisfy the requirements of the principle investigator.

2. Provide a basis for the daily planning during the actual mission, obviously this could not be exact since the real orbit and hence times of events differs from that predicated prelaunch. Also the radiators appear to be less efficient than predicted (this latter providing a benefit in power available as less heating is required to keep the freon loop inside its nominal operating temperature range).

Operational planning; This is carried out daily with the aim of generating a list of telecommands for uplink to the on-board master schedule for time-tagged execution. It takes account of the following:

- the baseline plan
- requested changes (from the principle investigators)
- latest orbital event file
- known spacecraft status (i.e. resource availability)

The EURECA mission planning system provides tools which assist both phases. These consist of the following:

1. Operation Definition Editor; this defines the resource profiles (power, cooling, data rate etc.) of each schedulable operation as a function of time.

2. Operation Request Editor; this is used to define the time constraints which should be applied when scheduling the start of an operation. These constraints are expressed as windows relative to GMT, orbit number, orbital events (eclipse start/end, South Atlantic anomaly transits, ground station passes) or combinations of these.

3. Mission Planning Aid (MPA); this is an interactive tool, with an X-windows based user interface. The operation definitions and requests defined by the above editors, along with the orbital event file, are used as input to schedule operations. It can be run in three basic modes. The first mode is fully interactive, in which the user must intervene to resolve any clashes found, either by accepting the clash, rejecting the operation or moving the start of the operation such that there is no longer a resource conflict. The other two modes are "semi-interactive" where the user can either specify that all operations causing clashes be rejected or alternatively be accepted.

These last two modes are described as "semi-interactive" because certain conflicts (viz the execution of master schedule commands during downlink of the onboard memory unit) are considered too critical to be handled automatically and so user intervention is always required in these cases.

The above tasks run on VAX workstations in complete "stand alone" from the rest of the EURECA dedicated control system (EDCS). Once the daily planning has been completed a file of commands for the desired time period, typically extending 48 hours from the end of the next pass sequence, is generated and then sent via DECNET to the prime EDCS VAX, where it is inserted into the commanding chain and later uplinked.

CRITICAL DISCUSSION

A principle difference between the EURECA mission planning approach and those of ERS-1 and ISO is the absence of an explicitly identified user-oriented step, despite the quite large number of PIs. The whole process takes place at the control centre. Because mission planning is done under a single responsibility this user-oriented step was not explicitly identified.

The general principles upon which the EURECA Mission Planning Aid is based seem to be sound. Much of the planning of the mission is done in advance. The bulk of the changes in the schedule from day to day, will be due to the shift in the orbital event times. Inputs from PIs also influence the daily schedules, but have not been, so far, large in volume: The mission design, in which prefiltering and harmonisation of requests was done pre-launch has been borne out in practice. However, such an approach would not have been appropriate for a mission like ERS-1, in which the influence of user requests is much more dynamic and cannot be planned in advance to the same extent.

The MPA detects resource clashes. However, the following deficiencies should be noted:

- it does not give an analysis of the contributions to the resource clash; to find this out the user has to examine the operations requests, which is not straightforward especially since instruments switched on considerably earlier than the time of the clash may be resource consumers.
- it does not provide any means to automatically resolve resource clashes (as the ERS-1 MPS does).

This reflects the fact that PIs have not been able to specify any algorithm for solving these clashes.

- it lacks flexibility in applying the scheduling constraints. The requirements for these were defined well before launch, but in practice it has been found that change requests can be difficult to formulate in terms of the scheduling rules.

- The command request interface with the PI's are received in a written form. They can be slightly "fuzzy", which involves the operations staff in considerable interpretation work.

V. LESSONS LEARNED

Examination of the critical discussions above, shows the following common themes:

1. Mission planning systems naturally split into two parts, the user interfacing part (Phase 1 or MPP1) and that producing the operations schedules and interfacing to the control system (Phase 2 or MPP2). A number of major problems resulted from difficulties in the set up and interaction of these two parts, particularly when they are under separate responsibilities. Problems can be avoided by clear formal agreements on separation of tasks, document and controlled interface descriptions/agreements.

2. Mission Planning systems carry out their scheduling using a set of rules and constraints. Problems result both from late knowledge of these rules and from the "hardwiring" of these rules into planning systems, making the inevitable changes and corrections difficult.

3. Mission planning systems normally run off-line; as a consequence they tend to be tested less thoroughly in operations than spacecraft control systems. However they are already important operational tools and will become more so as missions become more complex. Testing of mission planning systems will therefore have to be more thoroughly integrated into mission test programs.

4. The discipline of goal-oriented optimisation needs more thorough exploration, at least as far as ESA Mission Planning systems are concerned. Algorithms from the Operations Research disciplines exist to do this, but there has been little practical study of their application to Mission Planning.

5. Projects are being set up in which the division of responsibilities (a management matter) assumes that platform and payload control can be separated (a technical matter). The ISO example is a more or less successful attempt to do this, although it does illustrate some of the potential pitfalls of this approach.

6. While certain aspects of mission planning are of a specialised nature (perhaps mission specific) it is possible to identify areas (e.g. scheduling rules) where it may be possible to use existing commercially available products to reduce the amount of customised software.

VI. GENERIC MISSION PLANNING TOOLS

Having completed this rather high level review of some recent mission planning systems developed at ESOC, it is worth noting that all the systems were developed independently and use no common elements. While it is clear that there are a number of essential differences between the missions concerned, it is also clear that there are some commonalities. Is it possible to identify some common functions which could be implemented as reusable tools? An ongoing ESA study is examining this problem. To date the mission planning approaches discussed here have been analyzed in detail, together with a number of other approaches from both ESA and NASA. 31 distinct mission planning functionalities have been identified and a commonality matrix established for the various missions. Common functions include, among other planning request handling, orbit propagation, activity scheduling, conflict identification and resolution, time-line visualisation with zooming, command schedule generation. The plan is that a workable concept for generic mission planning facilities can be developed and that key elements of it can be prototyped and demonstrated in application to real missions. This should feed into both the ongoing programme of spacecraft control infrastructure development (SCOS-II, see Ref. 2) and the Advanced Technology Operations Systems (ATOS, see Ref. 3), which seeks to develop knowledge-based elements to be added to this infrastructure.

VII. REFERENCES

1. Mission Planning for an Earth Observation Low Earth Orbiter: ERS-1, K Hirst and E M Sørensen, 1992, Proceedings of SPACEOPS 1992, Pasadena, CA,
2. SCOS-II: ESA's New Generation of Mission Control System, 1992, J F Kaufeler et al, Proceedings of SPACEOPS 1992, Pasadena, CA,
3. The Advanced Technology Operations System ATOS, 1992, J F Kaufeler et al, Proceedings of SPACEOPS 1992, Pasadena, CA.

THE MARS MAPPER SCIENCE AND MISSION PLANNING TOOL*

Martin W. Lo **

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California, USA

ABSTRACT

The Mars Mapper Program (MOM) is an interactive tool for science and mission design developed for the Mars Observer Mission (MO). MOM is a function of the Planning and Sequencing Element of the MO Ground Data System. The primary users of MOM are members of the science and mission planning teams. Using MOM, the user can display digital maps of Mars in various projections and resolutions ranging from 1 to 256 pixels per degree squared. The user can overlay the maps with groundtracks of the MO spacecraft (S/C) and footprints and swaths of the various instruments on-board the S/C. Orbital and instrument geometric parameters can be computed on demand and displayed on the digital map or plotted in XY-plots. The parameter data can also be saved into files for other uses.

MOM is divided into 3 major processes: Generator, Mapper, Plotter. The Generator Process is the main control which spawns all other processes. The processes communicate via sockets. At any one time, only 1 copy of MOM may operate on the system. However, up to 5 copies of each of the major processes may be invoked from the Generator.

MOM is developed on the Sun SPARCStation 2GX with menu driven graphical user interface (GUI). The map window and its overlays are mouse-sensitized to permit on-demand calculations of various parameters along an orbit. The

program is currently under testing and will be delivered to the MO Mission System Configuration Management for distribution to the MO community in 3/93.

Key Words: Mission planning software, digital map display, groundtrack, instrument swath

1. INTRODUCTION

The Mars Mapper Program is an interactive tool for science and mission design developed for the Mars Observer Mission. MOM is a function of the Planning and Sequencing Element of the MO Ground Data System. The following is an overview of the MO Mission to provide the context in which the Mars Mapper is used.

The goal of the MO Mission is to study the martian surface, atmosphere, and gravitational and magnetic fields. To achieve these scientific objectives, the MO S/C will be placed in a sun-synchronous orbit at roughly 400 Km altitude during the Mapping Phase. This is a near-circular, near-polar orbit with repeating groundtracks typical of mapping missions. The MO S/C is currently on its way to Mars and will reach Mars in August of 1993. The Mapping Phase will begin in 12/93 and will last approximately 2 years.

The MO science payload consists of 7 instruments and the Radio Science Investigation (RS). The instruments are: the Gamma Ray Spectrometer, the Magnetometer/Electron Reflectometer, the MO Camera, the MO Laser Altimeter, the Pressure Modulator Infrared Radiometer (PMIRR), the Thermal Emission Spectrometer (TES), and the Mars Balloon Relay. During the Mapping Phase, the MO S/C is nadir pointed. With the exception of PMIRR and TES, the MO instruments are not pointable. Both PMIRR and TES use scanning mirrors to alter the pointing of

* The software development described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

** Member of Technical Staff, Launch Earth Orbiting Mission Analysis Group, Mission Design Section, Member AIAA

the instrument view. PMIRR is able to point along-track as well as across-track. TES is able to point along-track only. Both instruments are used to point at the planet limb for atmospheric analysis.

2. FUNCTIONAL REQUIREMENTS SUMMARY

The functional requirements for MOm are described in Sec. 4.2.4.11 of the MO Mission Operations Specification Document, Vol 2 (Ref. 1) and are summarized here:

- ◆ MOm shall provide instrument and mission geometric data generation capabilities to support the science mission planning and sequencing activities of the MO Project.
- ◆ MOm shall generate trajectories, instrument footprints, and planetary and solar geometry-dependent information to be displayed in text and in graphical formats as hardcopy or electronic displays.
- ◆ MOm shall read the JPL Navigation Ancillary Information Facility (NAIF) Kernel files and use the NAIF Toolkit to generate all ephemeris and instrument geometry parameters.
- ◆ MOm shall display mission geometric data as vector overlays on digital mars maps produced by the US Geological Survey.
- ◆ MOm shall plot mission geometry data as XY-plots.

Although a GUI is not required by the original functional requirements, due to the graphical nature of MOm and the availability of GUI builders, a GUI was added to the functional design. Refer to the MOm Functional Design Document (Ref. 2) for additional information on the functional requirements and design of MOm.

3. PROGRAM DESCRIPTION

The Mars Mapper assists the scientists and mission planners of the MO Mission by providing both visualization of and values for various mission parameters and instrument geometry parameters during the orbital Mapping Phase of the MO Mission. The user interacts with the MOm through the GUI which is completely menu-

driven. MOm consists of 3 major processes: the Generator, the Mapper, and the Plotter. The Generator is the main control process which spawns all other processes and is the engine which produces all of the mission data of which there are two classes: overlays and parameters. The overlays are data to be displayed by the Mapper such as the S/C groundtrack, instrument footprints and swaths, day/night terminators, planet limb at a point along the orbit, sites, orbit events along the orbit such as the rise/set of the earth, sun, and DSN stations. The parameters are the mission parameters generated at each time step such as sun angles, S/C range, the local hour angle, etc. which can be plotted against time or one another by the Plotter. Close to 200 mission parameters are available to the user. The parameters are selectable by the user and only those selected by the user are computed, displayed, and sent to the Plotter.

The Mapper and Plotter are the visualization tools of the Mars Mapper Program. The Mapper can display digital maps in various projections and overlay the digital maps with groundtracks of the S/C or instrument footprints and swaths provided by the Generator. The overlays are grouped into segments which can be selectively erased, plotted, saved or deleted. This approach is derived from the segment concept of the GKS graphical standards. Similarly, the Plotter displays XY-plots of parameters produced by the Generator. The data generation and display are interactive: the data is displayed by the Mapper and the Plotter as soon as it is computed by the Generator. The plot data may also be directed to a file by turning on a Writer in the Generator. Files created by the Writer are readable by the Plotter for XY-plots.

MOm also has an "Interactive Mode" which sensitizes the mouse to the S/C groundtrack on the Mapper. Depending on the options selected, by clicking the left mouse button at a point along the groundtrack, MOm can display the instantaneous day/night terminator, limb, and the mission parameter values at that point. Thus the user may examine the parameter values along the orbit interactively.

Figure 1. below is the main menu of the Generator Process. The menu bar at the top provides pull down menus which allows the user to set the

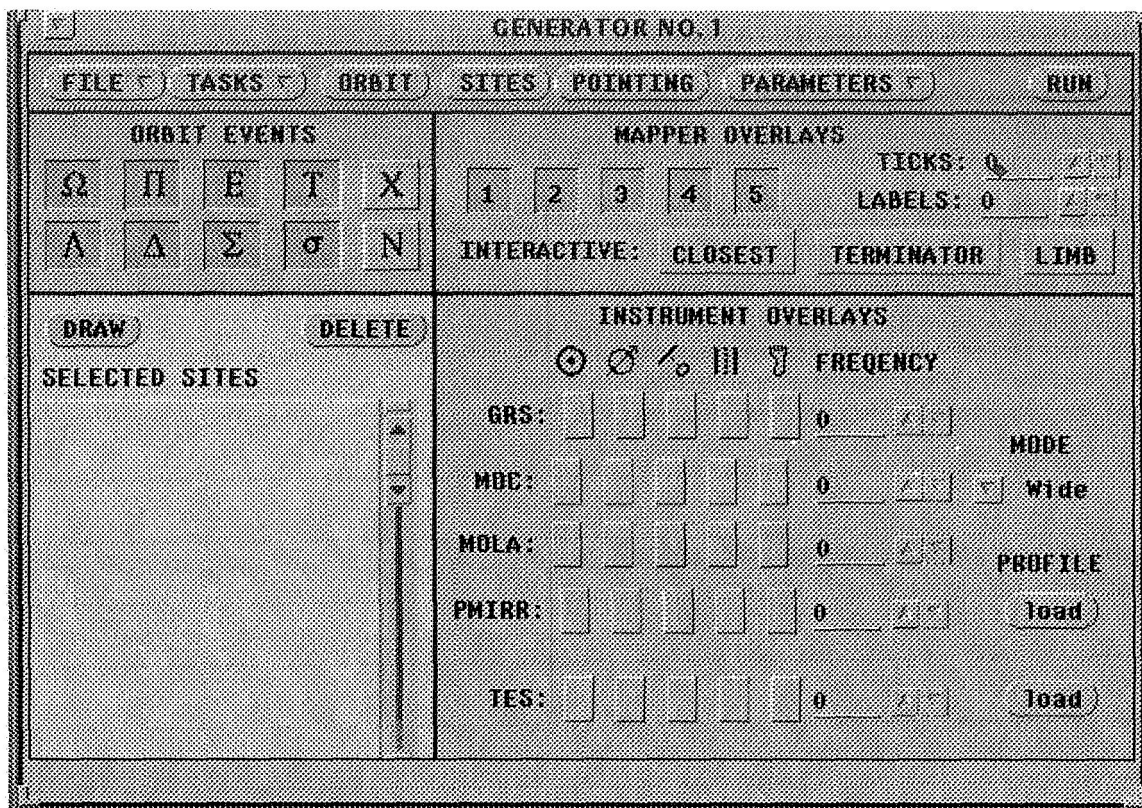


Figure 1. Generator Menu

orbit, invoke Mapper, Plotter, Writer and additional Generator tasks, load and search site databases, analyze instrument pointing, and select and display parameters. The rest of the menu lets the user define the overlay interface between this Generator and the various Mappers. The ORBIT EVENTS Panel at the top left corner just under the menu bar allows the user to select orbit events to be plotted along the groundtrack on the Mapper. The SELECTED SITES List just below the ORBIT EVENTS allows the user to display or delete sites on the Mapper. The MAPPER OVERLAYS Panel lets the user connect this Generator to 5 different Mappers. Here is where the INTERACTIVE Mode buttons are set to sensitize the mouse to the groundtrack for the display of instantaneous terminators, limbs, and parameter values. The INSTRUMENT OVERLAYS Panel allows the user to customize the instrument overlays.

4. ARCHITECTURE AND OPERATIONAL CONCEPT

The 3 major processes of the MOm are the Gen-

erator, the Mapper, and the Plotter. The Generator is the main control which spawns all other processes. The processes are independent and communicate via sockets which provide very efficient two-way interprocess communications. The main advantage of this design is that both the Mapper and the Plotter are independent of the Generator and of the MO Mission. Thus, these programs can be used for other projects without modification. By loading earth maps, the Mapper can display earth orbits. By loading star maps, the Mapper can display orbits on the celestial sphere for skyward pointing missions. In particular, the user may develop his/her own programs to display his/her data in conjunction with that generated by the Generator. Since each Mapper/Plotter is uniquely identified by its socket port number, any process on the network can communicate with the Mapper/Plotter to display data.

This design also enables the use of multiple copies of the Mapper or Plotter. At any one time, only a single copy of MOm may operate on the system. But, within each MOm, the user can in-

voke multiple copies (up to 5 each) of the Generator, Mapper, and Plotter. For example, if the user wishes to have a global view and a close up view of an orbit track, this can be achieved with 2 Mappers connected to a single Generator. Or, if the user wishes to compare the groundtrack of orbits separated by 100 days, this can be achieved by using 2 Generators plotting to the same Mapper. However, the program is not designed to operate at full capacity where all possible processes are invoked. This will push the system resources beyond their limits. The intent is to give the user as much flexibility and control as possible and let the user manage the system resources.

The socket interface to the Mapper is very simple; all Mapper overlays are drawn by using 5 Mapper commands: LINE, TEXT, TICK, SYMBOL, INPUT. The first four are self evident, the last command, INPUT, enables the user to input Mapper commands stored in files. The commands have various parameters which are described in Sec. 7 of the MOM User's Guide (UG, Ref. 3). Each command can also be given a segment name of an arbitrary ASCII string. This enables the Mapper to group the commands into segments which can then be manipulated. This feature enables the user to selectively erase or delete, save and reload overlays.

Similarly, the socket interface to the Plotter is also very simple. The Plotter expects a list of parameter names to be followed by the data in column form. The exact format is described in Sec. 8 of the MOM User's Guide (Ref. 3).

Typically, when the user runs MOM, the Generator Menu (Fig. 1) pops up. The user sets the orbit parameters and the start/stop times and step size under the ORBIT Button. The user may turn on a few instrument footprints or swaths. Then the user brings up a Mapper, Plotter, and Writer under the TASKS Button. The user needs to load a digital map into the Mapper and specify the parameters to be generated under the Generator PARAMETERS Button. Then the user must set the Plotter to plot the parameters just selected. Finally, when every thing is set, the user clicks the RUN button and data generation begins. Groundtracks and instrument tracks appear on the Mapper and plots appear on the Plotter. The socket interfaces are completely transparent to

the user who notices only the different windows popping up processing and displaying the data as he/she specified.

This architecture separates the mission analysis code from the graphics code. This has several advantages. Clearly, the mission-independent graphics code can be reused as-is for any other mission. Completely separating these two very different tasks requiring very different skills, simplifies the software management and maintenance. However, success depends on a simple and well defined interface between the processes.

5. SOFTWARE DEVELOPMENT METHODOLOGY

The MOM software development follows the JPL software standard practices. This includes the functional design review and the combined software requirements and specification design review. Normally, the requirements design review and the specification design review are separate. Due to resource constraints, a waiver was obtained from the project to combine these reviews. For the same reasons, peer review and inspection were not instituted. The resultant requirements and design are documented and described below.

During the functional design phase, the goal was to understand what the MOM does without specifying how the MOM performs these tasks. External interfaces were identified, functional blocks were identified, some new requirements were added and obsolete requirements were deleted. This is documented in the MOM Functional Design Document (Ref. 2).

During the software requirements and specification development phase, the goal was to describe in greater detail what the MOM does. Emphasis was placed on the architectural design and the external interface definition. Extensive functional and data flow diagrams were constructed to review the software architecture and design and to identify possible flaws. At the same time, the GUI was being built. Since there is a close correlation between the various menu items and functionality, the GUI was a design driver and turned out to be an excellent design tool. The resulting

design is documented in the MOm Software Design Document (Ref. 4). Coding began early in the software requirements/specification phase due to schedule constraints. The test version delivery for October 1992 was met.

6. OPERATING ENVIRONMENT

The Mars Mapper Program Version 1.0 has been developed on a Sun SPARCStation 2GX under SunOS V. 4.1.2 and Openwindows V. 3.0. The following programming languages and tools were used for the development:

F77	V. SC1.0
C	SunOS V.4.1.2
DevGuide	V. 3.0
XView	V. 3.0
Codebase	V. 4.2
NAIF Toolkit	V.17.2
PDS Toolkit	
PGPlot	V.4.9

Due to resource constraints, a waiver was obtained from the project to use Sun's Openwindows instead of MOTIF which is the MO Project standard window manager. We had inherited code from previous prototypes which used DevGuide and XView. At the time, the MOTIF GUI builders were expensive and not as easy to use as was DevGuide which was free. We also had little experience with MOTIF and could not have met our tight schedule with the additional time required for the MOTIF learning curve. For additional information, refer to Appendix B of the MOm Functional Design Document (Ref. 2).

The Mars Mapper should be operated on a color workstation. However, it will function with a black and white monitor where the digital map displays are dithered. It can also be brought up under MOTIF although some of the Greek fonts used are replaced by Roman fonts under MOTIF.

7. APPLICATIONS

Although MOm was developed with science and mission planning in mind, it can be equally useful for the analysis of actual and archived data so long as the necessary ephemeris data is in the

NAIF format. The Mars Mapper can also be easily adapted to other planetary missions. Although the instrument footprints and swaths are specific to the MO mission, they can be readily changed to accommodate other missions. The Mapper and Plotter are completely independent of the Generator and the MO mission; they are generic tools. By changing the digital map, the Mapper can display missions for any other planetary body. In fact, we have been using the Mapper to analyze observatory missions such as SIRTf and COGEX. Thus, if a new Generator needs to be developed from scratch, the visualization tools are already available.

8. ACKNOWLEDGEMENT

This work was done jointly with Min-Kun Chung, the Cognizant Programmer for the Mars Mapper Program. The author also wishes to thank Robert Mitchell and David Farless for their helpful suggestions.

9. REFERENCES

1. Hammer, F. et al. 1991, Mars Observer Mission Operations Specification Volume 2: Data System, Launch Phase Final, JPL 642-315
2. Lo, M. 1991. Mars Mapper Functional Design Document, JPL 642-3501-MAP-E.
3. Lo, M. et al. 1992. Mars Mapper User's Guide Version 1, JPL 642-3505-MAP-E.
4. Lo, M. 1991. Mars Mapper Software Design Document, JPL 642-3511-MAP-E.

MISSION PLANNING FOR AN EARTH OBSERVATION
LOW EARTH ORBITER: ERS-1

N 9 4 - 2 3 8 6 8

Paul Lockyer

Cray Systems, Transom House, Victoria Street, Bristol, UK.

ABSTRACT

ERS-1, the first European Remote Sensing satellite, has a payload predominantly of microwave instruments and is in a polar sun-synchronous orbit.

All ground and on-board activities from user requests to delivery of data products are combined into one integrated system. In view of the high number of products which can be generated by ERS-1, the Mission Planning System (MPS), which plans the on-board activities of ERS-1, is an essential tool for operations since manual planning of the large number of daily operations is out of the question both on workload and safety grounds. In addition the MPS, in line with the integrated nature of the ERS-1 system, also plans activities at the prime ground station, including among others, the operation of the payload data processing systems there.

This paper outlines the operations concepts for ERS-1 mission planning, and describes the Mission Planning System used at the ERS-1 Control Centre. Novel functionalities, such as automatic resource clash resolution, are described. A critical discussion gives lessons learned for future mission planning systems.

Key Words: Mission Planning, Resource scheduling, Expert System, ERS-1.

1 INTRODUCTION

A mission of the range and diversity of ERS-1 requires careful operational planning in order to make efficient use of the opportunities for data acquisition during the limited mission lifetime. The Mission Planning System (MPS) is required to produce the command schedules for the spacecraft, acquisition schedules for the ground segment and processing schedules for the prime ground station, subject to a large number of operational mission constraints.

1.1 The ERS-1 Mission

The ERS-1 satellite is in a sun-synchronous polar orbit of the Earth of approximately 100 minutes duration ; 70 minutes of exposure to the sun and 30 minutes of darkness. The ERS-1 mission provides data which are used to address a wide range of environmental issues as well as adding a new dimension to our studies of the Earth.

1.2 The ERS-1 Spacecraft

The ERS-1 spacecraft is based on a SPOT platform with the scientific instruments and the on-board Instrument Data Handling and Transmission (IDHT) system controlled by MPS generated command schedules.

A brief summary of these instruments and their operating constraints shows some of the requirements of the MPS instrument resource conflict checking algorithms.

1.2.1 The Active Microwave Instrument

The AMI contains two separate radars and can be operated in one of four active modes - Synthetic Aperture Radar (SAR) image mode, wind, wave or wind/wave interleave mode. Each of these active modes has various different categorisations dependent upon calibration; on-board or on-ground range compression and whether the satellite is in roll/tilt mode or not.

Any switching between active modes can only be achieved by first switching to one of three inactive modes. Each mode has minimum and maximum durations for individual switchings. The image modes also have tolerances on the amount of time they can be operated in eclipse and daylight periods.

1.2.2 The Radar Altimeter

The RA is a nadir pointing pulse radar designed to make precise measurements of echoes from

ocean and ice surfaces. The RA can be commanded in one of six active modes with only one inactive mode. Operating constraints for the RA apply in a similar manner to those described for the AMI.

1.2.3 The Along-Track Scanning Radiometer and Microwave Sounder

The ATSR-M consists of two instruments, an Infrared Radiometer and a Microwave Radiometer, and is used to measure global sea-surface temperature for climate research purposes. The instrument is operated for long uninterrupted periods in a particular mode.

1.2.4 The Precise Range and Range-rate Equipment

The PRARE was to have been the simplest of all the instruments to command. Its internal computers gathering ranging data, storing it, and then dumping it independently of the main spacecraft IDHT system. Unfortunately, the PRARE has not yet been operational.

1.2.5 The Instrument Data Handling and Transmission System

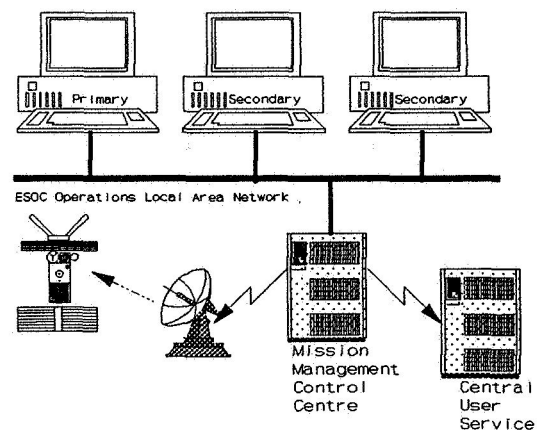
ERS-1 has two telemetry systems. There is the classical telemetry, tracking and control system operating at S-band. As this low rate (2Kbit/s) cannot be used for the science data the platform also includes a complex IDHT system. This system consists of two tape recorders, each having a 6.5Gbit capacity, which are operated redundantly and two X-band links. One X-band link is dedicated to the AMI SAR and transmits real time image data at High Bit Rate (HBR). The other link is used to dump recorded data, live instrument data and a copy of the S-band data at Low Bit Rate (LBR).

In addition to the timings of certain events which trigger record, playback and link switchings the system has constraints dictating that the independent units cannot be commanded simultaneously. When a recorder reaches beginning of tape it will automatically switch to a standby mode. The timings of these events must be predicted by MPS and the commanding of other IDHT units must not coincide with these times.

1.3 The ERS-1 Ground Segment

The primary ground station for ERS-1 is in Kiruna, Sweden. The station is remotely controlled by the Mission Management and Control Centre (MMCC) at the European Space Operations Centre (ESOC) in Darmstadt, Germany using MPS generated schedules. It contains the equipment to control and operate the satellite and the ERS PROduct DISsemination (PRODIS) system. PRODIS collects the science data and processes it into packaged data products. Many of these, the Fast Delivery Products (FDPs), are disseminated rapidly to users. Such products can be requested by users via the Central User Service (CUS).

2 SYSTEM ARCHITECTURE



System Architecture

The MPS resides in the Mission Management and Planning Office (MMPO) at ESOC. It consists of three VAX 3100 workstations. The VAXes are connected via DECnet on the ESOC operations Local Area Network to the MMCC from which it receives and transmits its input and output data. The system has been implemented primarily in Pascal with a DECwindows user interface.

3 PLANNING ENVIRONMENT

The MPS services user requests in the most efficient manner possible given numerous predefined constraints. These requests originate from Principal Investigators throughout the user community who interface directly with the CUS at the Earthnet ERS-1 Central Facility at ESRIN in Italy. Requests also come directly from the ERS-1 project at ESTEC in the Netherlands and from the

MMPO at ESOC.

4 SYSTEM INPUTS

4.1 Scenario Definition

In order to control the ERS-1 spacecraft the MPS requires the definition of all ground stations used during any orbit. This data is used by MPS to determine commanding windows and timings for down linking of HBR and LBR instrument data in addition to the normal spacecraft telemetry data. When performing a data dump to high-latitude ground stations such as the prime Kiruna station, the spacecraft solar array sometimes causes a brief occultation of the link. The occurrences and timings of these occultations are calculated within MPS and are used in the planning algorithms.

The ERS-1 project at ESTEC have defined zones covering all areas of interest on the Earth's surface. These zones cover areas of land, sea and ice. Each zone has a unique identifier and the MPS calculates timings of zone crossings by the spacecraft in a similar manner to the visibility of ground stations. The different instrument modes have different ground swaths which mean that the timings of zone crossings are different for each instrument and mode.

4.2 Instrument Requests

The ERS-1 MPS accepts requests for instrument switchings via four different mechanisms which are described below :

4.2.1 Instrument Zone Operation Requests (IZORs)

IZORs require individual instruments to be in particular modes whenever the relevant swath enters a particular zone. This is the simplest method for the project to ensure that land data is acquired when the spacecraft is over land and ocean when over ocean etc. IZORs have a priority and start and end validity date and time.

4.2.2 Sensor Operation Requests (SORs)

MPS also accept SORs which are explicit requests for individual instruments to be in particular modes at given times. These requests also have an associated priority.

4.2.3 Operations Requests (ORs)

It is possible for the MMPO to create ORs. These requests are typically for specific spacecraft command sequences, defined within the MMCC, but which require some, or all, of the instruments to be in a predefined state before and after the command is issued. The requests also have an associated priority as well as a user defined scheduling mechanism.

4.2.4 Emergency Requests (ERs)

In the event of a computer link failure between ESOC and the CUS at ESRIN in Italy, user requests can still be entered into the system manually by the MMPO staff using ERs. These requests have the same attributes as SORs though the source for these requests can come from fax, telex or normal telephone messages.

4.3 Fast Delivery Product Requests (FDPs)

Users can request processed product items from the Kiruna PRODIS system. These requests contain start and stop times as well as product type and other necessary attributes for each FDP.

4.4 Flight Dynamics Corrections

The MPS utilises a reference orbit model. It is inevitable that the orbit will drift slightly from this reference and indeed fine orbit control manoeuvres are made at regular intervals to correct these drifts. In order to accurately command the scientific instruments MPS uses data provided by the Flight Dynamics Services team at ESOC who are responsible for the attitude and orbit control of ERS-1. The actual UTC times for the start of each orbit are corrected by MPS daily according to this input data.

4.5 Unavailability Data

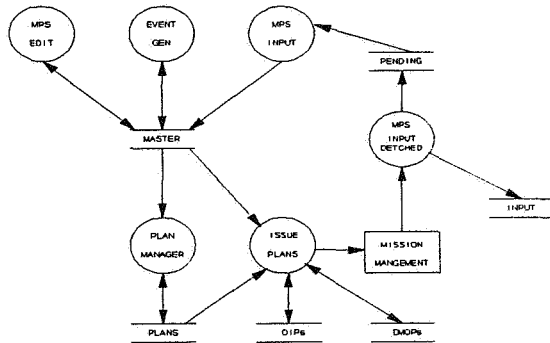
MPS also accepts input data detailing ground station and PRODIS equipment unavailability timings. It is necessary to schedule alternative ground stations or use alternative processing equipment to generate and disseminate ERS-1 data products if such data is received.

5 SYSTEM OUTPUTS

In a standard planning cycle the first system output is the Detailed Mission Operations Plan (DMOP). This plan, detailing spacecraft operations, orbital events, zone and station visibility data and fast delivery products, is returned to the CUS at ESRIN providing feedback as to which original requests will be satisfied. This data is usually presented to CUS three weeks before the planned events.

Once the DMOP has been agreed the Operations Implementation Plans (OIPs) are generated by MPS. These files, including the actual command schedules for the spacecraft and PRODIS, are generated the day before the scheduled execution time in order to have the latest, and most accurate, flight dynamics corrections.

6 DATA PROCESSING



Data Flow Diagram

The MPS system consists of five interactive processes and one detached process. Most of the processes are used to control the configuration of the many entities involved in the planning exercise. The heart of the MPS is the planning executive, the Plan Manager, which is used to bring all the configured input items together, schedule the applicable requests according to the operational constraints, generate secondary events such as tape recorder switchings, which are entirely dependent on instrument activities, and produce a data file from which DMOPs and OIPs can be produced.

The planning routines operate on Pascal data structures based time and event classification ordered linked lists. All MPS data flow is via binary files which are based on this data structure.

6.1 Mission Management

Mission Management is an external process, part of the ERS-1 MMCC, running only on the primary node. This process provides the input data for planning and accepts the output data files from the MPS for distribution to the relevant command and control centres.

6.2 MPS Input (Detached)

This process is the MPS interface with Mission Management. No operator interface is required as this automatic process receives all input data and converts it into an internal binary format for later processing. The original text files are also maintained on disk for operator reference.

6.3 MPS Input

An interactive process providing the operator with an interface through which to view, accept or reject any of the binary files created by the MPS Input (Detached) process. Any files which are accepted will be merged into existing master database files to be used within the MPS.

6.4 Event Generator

The Event Generator is used to create, modify or extend a Reference Orbit Template (ROT).

The ROT contains data pertaining to the quasi-static events occurring during each orbit. The events described include orbit start times, entries into and out of eclipse, times of ground station visibility and the times of entries and exits of the different instrument swaths over the predefined zones.

6.5 MPS Edit

The MPS Edit facility provides the user with an interface to the orbit definition, input data and the ROT master database files permitting manual edit facilities.

6.6 Plan Manager

The Plan Manager process contains the planning executive and is an interactive process giving the user some influence over the processing of all input data and the generation of binary database files consisting of the actual data which is used

to generate the DMOPs and the OIPs.

The planning executive starts by taking a snapshot of the ROT for a given time period, measured in orbits. The user must also enter the start planning status vector defining items such as the start modes for each instrument.

6.6.1 Request Scheduling

The quasi-static data from the ROT is supplemented by adding data from the request master database files. Every request and its attributes will be added and marked as 'enabled'.

6.6.2 Primary Conflict Checking

Once all requests have been scheduled the conflict detection algorithms are applied. At this point conflicting requests for a given instrument are resolved on a priority basis. Any conflicting requests which are concurrent with a given high priority request are marked as 'disabled'. It is possible to have a single low priority request split into two or more segments because of overlapping higher priority requests.

6.6.3 Request Merging

The MPS is expected to optimise instrument switchings in order to obtain optimum performance from the delicate scientific electronics. If an instrument has two contiguous requests for the same active modes they can be merged into a single request. In some circumstances it is permitted to join two consecutive requests for the same mode even if they are not contiguous.

6.6.4 IDHT Scheduling

Once the requests have been scheduled and instrument switchings have been optimised the schedules used to drive the IDHT can be generated.

The first data to be generated is the selection of ground stations to receive the LBR and HBR science data. Any station with a coverage time greater than an allowed minimum will be acceptable to receive HBR data generated by the AMI SAR. The LBR station selection is slightly more complicated as there must be only one per orbit. The station with the longest coverage and

the highest priority is selected as long as this does not contravene other specified rules. Some stations cannot be used for two consecutive passes because of local operating constraints.

Once the ground stations have been selected the recording time (time between passes) and the associated replay times (time over a LBR ground station) can be calculated. It is possible that initially there will be insufficient time to replay all the data which is recorded. If this happens the margins allowed for data acquisition at the ground station are relaxed permitting more replay time (the recorders replay over 13 times faster than they record). If there is still insufficient time then the recording will be descoped. This descoping is based on Earth zones at a given priority. The original user request priorities have no influence at this time.

The amount of data recorded and associated playback time may be affected by occultations. The tape recorder will automatically suspend replay and rewind the tape a little before starting again after any occultation. MPS does not need to explicitly command this but it does need to be aware of exactly how much data is being played back and when the recorders are switching modes.

Once the amount of data to be recorded has been calculated the IDHT switchings can be derived. The recorder is turned on when necessary and switched to playback for the ground station pass. The LBR link is turned on over the selected LBR ground stations. The HBR link is turned on if there is AMI SAR data and the satellite is over the HBR ground station.

6.6.5 Secondary Conflict Checking

The remainder of the instrument constraint checking is performed after all events have been processed. Rules checking for minimum and maximum durations for each instrument mode are checked. The operation of the AMI SAR in eclipse is also checked against predefined time constraints. Requests will be descoped if they contravene any rules.

At this point in the planning executive requests may be descoped to permit mode transitions. Most instruments require that standby modes are used between active modes, until now the Plan

Manager has only concerned itself with the scheduling of active modes.

6.6.6 Energy Management

As ERS-1 is powered in daylight by its solar panels and by battery in eclipse there are limitations on the power available at any given time. Once all instrument switchings have been derived for the whole planning period the MPS energy management routines ensure that a maximum 28% battery Depth Of Discharge (DOD) is achieved per orbit, a maximum average 24% battery DOD is achieved over the last five orbits and that the energy deficit from the battery is zero when entering eclipse.

Should any of these parameters be exceeded the routines calculate the equivalent amount of science data, taking into account any IDHT switchings which may change, which need to be descope. The relevant requests are descope and all IDHT switchings are removed. The planning executive is then repeated to re-apply the new IDHT switchings and confirm that the operational constraints are still being followed before the power figures are checked again. On the second pass the data should pass the checks. The descope is deliberately pessimistic and it may take more than two passes before the power constraints are satisfied.

6.6.7 Instrument and PRODIS Commanding

Once all the instruments requests have been scheduled a single pass is made adding the relevant spacecraft macrocommands. Some instrument modes have implicit switchings from active to inactive modes after a particular duration, others need to be explicitly commanded between each active and inactive mode.

The PRODIS equipment at the Kiruna data processing centre is scheduled dependent on the explicit FDP requests, as well as default products, using the predicted output of the instruments and the visibility of the spacecraft at the prime ground station.

6.7 Issue Plans

This interactive process enables the user to generate any of the MPS output files from the output of the Plan Manager process. This process

also adjusts the timings of events according to the flight dynamics corrections.

7 LESSONS LEARNED FOR FUTURE MISSION PLANNING SYSTEMS

ERS-1 is by far the most complex satellite operated by ESA. As a consequence the MPS is the most complicated operational satellite planning tool designed and used by ESA. Although the mission operating constraints were defined, and the MPS was delivered before the launch date, it was always anticipated that these would evolve post launch. For example, the system flexibility permits users to change defaults for instrument modes or operating margins for the IDHT following analysis of operational data by the ERS-1 project. Major changes in operations such as the IDHT recorder descope over zones (the original requirement being to descope according to request priority) have involved software design changes.

Future mission planning systems should aim to be even more flexible. They should allow for the evolution of mission operating requirements with minimal knock-on effects. It should be possible to switch individual rules in or out and customise the system still using the expertise of a system design engineer but without the need for relatively major software updates.

8 ACKNOWLEDGEMENTS

The author wishes to express his thanks to the staff of the Data Processing Division at ESOC who helped make this paper possible and to Mike Jones for preparing the abstract.

N94-23869

INTEGRATING PAYLOAD DESIGN, PLANNING AND CONTROL
IN THE DUTCH UTILISATION CENTRE

T. J. Grant

BSO/Aerospace & Systems b.v.
P. O. Box 1444, 3430 BK NIEUWEGEIN, The Netherlands
Tel: ++31-3402 88888 Fax: ++31-3402 60577

ABSTRACT

Spacecraft payload design, experiment planning and scheduling, and payload control are traditionally separate areas of activity. This paper describes the development under Dutch Government contract of a prototype software tool - the Activity Scheduling System (ASS) - which integrates these activity areas. ASS is part of a larger project to build a Dutch Utilisation Centre (DUC), intended eventually to support all space utilisation activities in The Netherlands. ASS has been tested on the High Performance Capillary Electrophoresis payload. The paper outlines the integrated preparation and operations concept embodied in ASS. It describes the ASS prototype, including a typical session. The results of testing are summarised. Possible enhancement of ASS, including integration into DUC, is sketched.

Key Words: Spacecraft payload, design, planning, control, integration, Dutch Utilisation Centre.

1. INTRODUCTION

Spacecraft payload design, experiment planning and scheduling, and payload control are traditionally separate areas of activity. Software tools supporting one area are rarely integrated with tools supporting another area. The purpose of this paper is to describe the development of a prototype integrated software tool for designing scientific payloads, for planning and scheduling experiments using payloads, and for controlling payloads using the resulting schedules. The prototype - known as the Activity Scheduling System (ASS) - has been implemented under a Netherlands Agency for Aerospace Programs contract, as part of a larger project to build a pilot Dutch Utilisation Centre (DUC). DUC is intended eventually to support all space utilisation activities in The Netherlands. The

DUC user may be either the payload's Principal Investigator (PI) or its Facility Expert (FE).

ASS is being tested on the High Performance Capillary Electrophoresis (HPCE) payload, which is the standard case study for developing DUC-Pilot software. HPCE is a general analytical technique for separating molecules by transporting charged particles through an electrolyte fluid in a fine capillary under the influence of an electric field (Ref 1). The HPCE payload is designed to be used for a wide variety of experiments in microgravity conditions, covering physical, chemical and biological processes, as a flexible, multi-user measurement facility serving many experiments on-board Columbus and many experimenters, both in-orbit and ground-based. HPCE can also be flown on other spacecraft, including ballistic sounding rockets, unmanned satellites (eg Eureka), and the Space Shuttle. In ASS, the HPCE payload is currently modelled as 27 entity-classes and 52 relation-classes, resulting in three sets of planning operators for payload assembly, preparation and operation.

The paper outlines the integrated preparation and operations concept embodied in ASS. It describes the ASS prototype, including a typical HPCE session. The results of ASS testing are summarised. Finally, possible enhancement of ASS, including integration into DUC, is sketched.

2. CONCEPT

2.1 Columbus USO and DUC

ESA has identified the need for an organisation to coordinate the use of Columbus. Earlier this year an international team reported to ESA on the definition of

a User Support Organisation (USO) for Columbus. The Columbus USO they proposed is a hierarchy, headed by a single Payload Operations Control Centre (POCC). Below the POCC, there could be one User Support Operations Centre (USOC) per nation. Each PI would have a User Home Base (UHB), which would be connected by communications networks to his/her nation's USOC. Mirroring the ESA-wide (E-)USO there would be national USOs (N-USOs). Nations could choose to implement their USOC with UHB functionality for PIs without their own UHB facilities; the USOC-UHB combination would be known as a Utilisation Centre (UC).

The Dutch User Support Organisation (DUSO) is seen as The Netherlands' N-USO. The DUSO would have as its primary goal to maximise the scientific output of microgravity research in The Netherlands. The Dutch user support concept (Ref 2) is based on performing as much as possible of the experiment preparation and operation in the user's own laboratory. There would be a Dutch Utilisation Centre which would support selected USOC-level functions and which could also provide UHB-level functions for those users who do not have equivalent facilities in their own laboratories. The DUC should be mobile, eg for positioning at a PI's location during an operations campaign. At a meeting in March 1992, representatives of the Dutch user community agreed that - funding permitting - a phased programme should be started to realise the DUSO and its associated DUC.

In a linked initiative, a consortium of Dutch organisations and companies, led by the Nationaal Lucht- en Ruimtevaartlaboratorium (the Dutch National Aerospace Laboratory), has started the detailed definition of the DUSO. The development concept (Ref 3) distinguishes two approaches:

The 'top-down' or 'formal' approach, which would result in a DUC developed according to a standard systems development method.

The 'bottom-up' approach, which would begin with the choice of a candidate experiment, ie a case study. DUC functions would be demonstrated by re-using existing infrastructure and applications in a pilot DUC environment.

In mid-1991, the consortium decided to progress both approaches in parallel. The HPCE payload was selected as the case study, with development of the DUC-Pilot environment beginning in late 1991. The HPCE Case Study sub-project quickly resulted in a set of functional requirements (Ref 4) which served as an input to DUC-Pilot development. In the absence (in January 1992) of a space version, the consortium decided to baseline its activities on a commercially-

available HPCE instrument (Ref 5) designed for use in (ground-based) laboratories.

DUC-Pilot Development re-uses the following sub-systems developed by consortium members:

- A generic Man-Machine Interface (MMI) tool.
- A payload simulator (Ref 6).
- A Planning and Control (P & C) sub-system, which became the Activity Scheduling System.
- A communications network simulator.
- A multimedia Document Filing System.
- A Multimedia Telesupport System for multimedia communication between PI/FE and crew.
- An interface to the Columbus Utilisation Information System.

These sub-systems were demonstrated as stand-alone applications, together with a Beckman HPCE instrument, at the 1992 European Conference on the International Space Year in Munich, Germany. In Phase 2 of DUC-Pilot Development the sub-systems are being integrated, for completion in December 1992.

2.2 Experiment Life-Cycle

The USO Definition Team has defined an experiment life-cycle which can be used to decide where N-USO support could be provided to Columbus users. The highest potential for N-USO support is to be found in those life-cycle processes which are generic (ie non-discipline specific), which are not better done at an international level, and which are not closely tied to the scientific peer judgement system. Such processes include payload design, experiment preparation, planning, and operations (including re-planning and maintenance).

At present, these processes involve the PI in generating large amounts of documentation: requirements documents, design documents, operating procedures, plans, post-flight reports, and so on. The one matter that all PIs agree on is that they would prefer to spend their time and effort in doing science, not in completing documents. Therefore, one good way of providing user support would be to reduce the amount of non-scientific documentation that the PIs must produce manually.

Documents are largely a means of exchanging information across system boundaries, whether the systems are people, organisations, or computers. One approach to reducing the manual production of documentation would be to identify the systems and the interfaces between them. There are two categories of interface: internal and external interfaces. Within an

N-USO, the internal interfaces are between the different processes in the experiment life-cycle. These interfaces could be made more transparent by integrating the processes. External interfaces, eg from an N-USO to the E-USO, could be made more transparent by making the processes open systems. In other words, nationally-provided tools must be able to exchange data with similar tools used at the international level; this would require a set of data-exchange standards. An enabling pre-requisite would be an agreement concerning the experiment preparation and operations concept underlying the processes. This paper proposes such a concept.

2.3 Preparation and Operations

2.3.1 Approach

In the development of ASS, attention has been focused from the start on making the internal interfaces between the life-cycle processes more transparent. The approach has been to determine the types of data that flow between them. The type of data output by one process must match the data-type input by the next.

2.3.2 Brief Theoretical Review

A very brief review of control, planning, and design theory is needed here. The process of control is intimately bound to the operation of a system, such as a payload. General systems theory regards a *system* as a process which behaves by taking inputs from its environment, processing them, and sending outputs back to its environment. *Control* is present when a feedback loop is added in which the outputs are inspected and compared with some objectives, and the results are added to the inputs. Controlling consists of making changes in the state of the operating system in order to influence what will occur in future and when it will occur. I distinguish between the subsystem under control and the controlling subsystem. The literature on control systems theory is vast; (Ref 7) is a good introductory text.

I define *planning* as the process of selecting and instantiating actions from the set of all known feasible actions for the system under control and logically ordering them into a sequence that will, on execution, transform a given initial state of the system under control into a desired goal state. This definition is consistent with that used in Artificial Intelligence (AI) (eg see (Ref 8)). Note that I make no stipulations regarding the constraints used in selecting, instantiating, and ordering the actions. Planning is a general process which can be specialised according to the type of constraint used. According to this view,

scheduling is the specialisation with additional, time-related constraints. As defined, the planning process is a combination of *resource allocation* and *sequencing*.

There are four key points here. First, the planning process takes place before the actions are performed. The planning process does not have to be completed before planned actions can be performed. As soon as an initial portion of the plan has been generated, then performance of that portion can begin. Second, the product output by the planning process is a data-structure - the *plan* - listing the intended actions by name, together with the resources to be used in performing those actions. Very often, the plan is realised as a (human-readable) document, or some electronic analogue. The plan is the input to the control process; in AI jargon, the plan is *executed* by the control process. Third, planning takes as its inputs (descriptions of) the feasible actions, the initial state, and the goal. Most importantly, the planning process is goal-directed. Notably, the goal is expressed in terms of the state that is desired. Fourth, planning does not itself satisfy the goal, but is necessary for its satisfaction. Only when the plan is executed are the plan's goals satisfied. Unplanned (ie non-directed) action gives no guarantee that any goals will be satisfied.

Design can be seen as another specialisation of planning in which actions have additional, geometric constraints. The inputs to design are the system's mission and a set of production constraints, eg for materials and facilities. The output - the *design* - is a specialised plan, which specifies the component parts of a system (cf resources) and the actions required to construct or assemble them. The design is executed by constructing or assembling the component parts to become the system as designed. The as-designed system represents the resources to be allocated during planning, and, after construction/assembly and planning, becomes the subsystem under control.

Note that the output of design does not specify the set of feasible actions that may be performed when operating the subsystem under control. However, planning needs a set of feasible actions as a part of its inputs. Therefore, some process must intervene between the design and planning processes to transform the output of design into the input of planning. The intervening process will be named *action-set generation*, which will also have operating constraints as an input.

2.3.3 Current Practice

Currently, spacecraft design makes extensive use of CAD techniques and tools. The "Mission" input to design takes the form of a document known either as

the Spacecraft Users Manual (SUM), or as the Operations Requirements Handbook (ORH). The SUM/ORH is authored by the spacecraft manufacturer using conventional word-processing facilities. There is an ESA standard for the layout and contents of an ORH, and ESA is currently prototyping an expert-system-based tool to support the authoring of ORHs to this standard. The output of the design process takes the form of documentation, often paper-based.

The design process is divorced from planning and control, because action-set generation is done manually. In the Western world, the space industry practice is to combine action-set generation with the planning process. Action-set generation results in the production of spacecraft operating procedures for routine and non-nominal operations. These procedures can be seen as sequences of action-descriptions, ie generic plans. For older spacecraft, these procedures are also published as paper-based documents. Planning then consists of retrieving the procedure appropriate to the current spacecraft status, instantiating it, and passing it to the Spacecraft Control System for execution. The planning and control processes are only linked electronically in the most recent spacecraft projects, and then only by file transfer. On-line links between planning and control are under development for Columbus.

2.3.4 Integration Concept

The proposed integration of design, planning and control is diagrammed in Figure 1 by means of the SADT notation (Ref 9). For clarity, the control and resource inputs to the SADT processes have been omitted. In the SADT notation, control inputs enter an SADT process from above, and resource inputs enter an SADT process from below. For example, the design output from the design process would also be an input to another SADT process (named "production", say). The output of the production process would be the subsystem under control, and this would become a resource input to the Control (& operation) SADT process. Moreover, each of the SADT processes could be supported by a tool, eg a CAD package for design, an action-set generation tool, a planning and scheduling tool, and a control system. These tools would have to be shown as resource inputs, and additional SADT processes would have to be added to represent the tool development processes.

There is one refinement not shown in Figure 1. Currently, spacecraft designers produce their designs in the form of documents, albeit aided by CAD tools. Later, the design document, together with the manually-produced operating procedures, would be used as the inputs to the development of a software simulator used by spacecraft operators for training and

contingency recovery. In the proposed preparation and operations concept, the development of a simulator would come first. The designers would work with a simulation-authoring tool to produce their designs in the form of a software simulation. They would explore the behaviour of this simulation, adjusting it as necessary. When satisfied with the result, the designer would trigger the tool to generate the design documentation automatically. The generation algorithm would be designed to produce the document according to the appropriate standards. This principle of automated generation of documentation would be repeated in the action-set generation, planning, and control processes. Although there would be little saving in the amount of documentation produced, the PI/FE would be spared the effort of creating and structuring documents. Moreover, consistency between documents and simulator would be ensured.

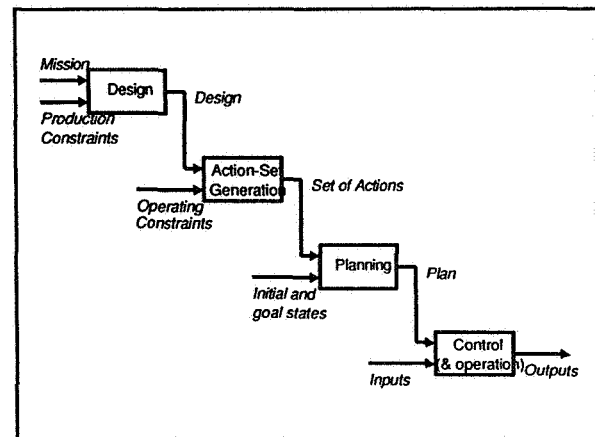


Figure 1. Integrating Design, Planning and Control.

Automatically-generated documentation has the further advantage that it has a regular structure that could be parsed by another algorithm, enabling automation of its input by another tool. In essence, such documentation is readable both by humans and by machines. For example, a design document generated automatically by the simulation-authoring tool could then be automatically input to an action-set generation tool.

3. INTEGRATED TOOL

3.1 Implementation Status

The ASS prototype has been implemented in Smalltalk/V on 386-class PCs under MS-DOS. The intended coverage and implementation status of the ASS prototype can be seen from Figure 2. The experiment life-cycle is shown in summary form from

left to right, with the processes covered by AI planners and schedulers shown beneath it. The dashed parts of the range for ASS show the functionality that, at the time of writing (November 1992), has yet to be implemented.

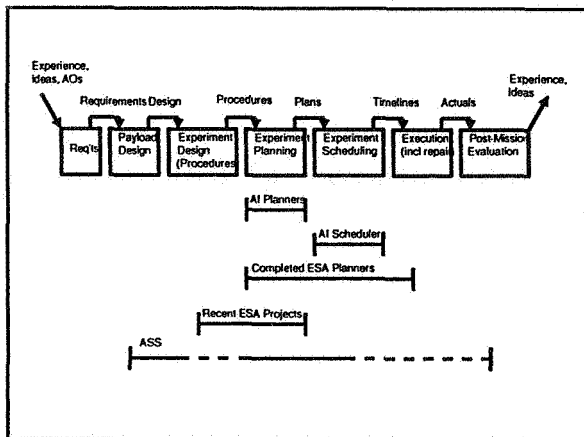


Figure 2. Role of Activity Scheduling System.

The implemented functionality is as follows. ASS supports payload design by representing it as an Entity-Relation model (Ref 10), extended with domain constraints. The user lists the classes of entities in the payload domain, the relations between those classes, and the instances of each entity-class. ASS then prompts the user for the constraints between relation-pairs. From these inputs, ASS generates a payload simulator whose behaviour the user can explore. When satisfied, the user can instruct ASS to induce an STN from the payload simulator. This induction process is computationally intensive. When the STN has been induced, the user can associate the use of external resources with each state and transition. Payload operating plans, together with summarised resource-usages, can be extracted from the STN for user-designated start and end states. At any time, the user can trigger the generation of a detailed design document. In addition, the user can initiate the automated generation of an ASCII file for input to an object-oriented analysis (OOA) tool which supports the Coad and Yourdon OOA method (Ref 11). Alternatively, the user may load the lists of entities and relations from an ASCII file prepared using the OOA tool.

The functionality still to be implemented is as follows. State- and transition-classes will be generalised from the STN, with transition-classes being represented as AI planning operators. (This functionality has been proven in another application.) Selected plans will be generalised as procedures. Schedules, timelines, and resource profiles will be obtained from selected plans by allocating external resources, as required by the parent state- and

transition-classes. The schedules or timelines will be ranked according to resource usage. Execution of the plans or re-instantiated operators and procedures will be modelled on the payload simulator. If the simulator is set to an invalid state, ASS will automatically detect this, and recommend a recovery plan (if one exists). If a recovery plan does not exist, this would indicate that one or more of the domain constraints had been violated. In this case, ASS would identify the set of violated constraints, modify the payload simulator, re-induce the STN, extract the new recovery plan, and execute it.

3.2 Technical Issues

ASS makes extensive use of AI algorithms, set in an object-oriented framework. Expert systems, AI planning, Truth Maintenance Systems, and machine induction all have a role to play. Domain constraints are represented as production rules, and constraint violation will be detected by forward-chaining inference. STN induction is performed using a variant of the *version spaces and candidate elimination* algorithm (Ref 12), with post-processing to identify valid transitions. No-good sets are obtained from the domain constraints to use in candidate elimination.

A major issue in ASS development has been to counter the combinatorial explosion in size of the version space during STN induction. Version space construction could be guided by heuristics, but only by sacrificing domain-independence and the guarantee of completeness. Object-oriented concepts have been more effective. The class-subclass relation, with inheritance, has been applied to entities to reduce the number of relation-classes needed to represent a given design. The class-instance relation has been exploited in partitioning entities, relations, constraints, states and transitions. Message-passing will be used to implement the payload simulation. The whole-part relation has a potential role to play in representing a payload as an assembly of sub-systems, but has not yet been incorporated in ASS. Despite these measures, it is recognised that STN generation remains NP-hard. There will still be some payload designs which are inherently so under-constrained that manual or heuristic methods will have to be used.

4. TEST RESULTS

Several toy domains are being used for development purposes, including the Dining Philosophers problem, a domain consisting of tanks and reactor-vessels (modelling fuel cells and refineries), and many versions of the blocks world. These domains vary from 2 to 6 entity-classes. Testing

with the HPCE domain has begun. The OOA tool has been used to construct a model consisting of 27 entity-classes and 52 relation-classes. (The same model was used in constructing the HPCE Payload Simulator (Ref 6)). This model has been loaded into ASS and a detailed design document generated. Two hours user-interaction was needed to determine the inter-relation constraints. Translation of the 1-to-1, 1-to-many, many-to-1, and many-to-many constraints captured by the OOA tool into ASS production rules would reduce this time substantially; this is being implemented. To date, only STNs for subsets of the 27 entity-classes have been induced. An algorithm for merging these "sub-STNs" into a complete STN is being designed. Hand simulation suggests that 20+ planning operators will result.

The existing procedure-oriented knowledge of operating the Beckman instrument is limited to a single procedure: the calibration of a newly-fitted capillary. The only existing "plan" is an abstract one shown on the instrument's front panel: RINSE, INJECT sample, SEPARATE sample constituents, and RINSE again. The instrument's manual states that empirical methods must be used to design experiments. Hence, any additional plans or procedures generated by ASS will be totally novel. We anticipate that the generated plans, procedures and planning operators will fall into three groups: instrument assembly (as performed in manufacture), experiment preparation, and experiment execution.

5. CONCLUSIONS

This paper has described the development of a prototype software tool which is designed to integrate spacecraft payload design, experiment planning, and experiment control. Traditionally, these have been separate areas of activity.

6. REFERENCES

1. Eckhard, F. 1992. High Performance Capillary Electrophoresis in the Microgravity Environment, *Advanced Space Research*, 12, 5: 247-255.
2. Visser, F.B. 1992. Dutch User Support Organisation Concept Description and User Requirements Document, NLR CR 92097 L, final issue, 16 Mar 92, Nationaal Lucht- en Ruimtevaartlaboratorium, Amsterdam, The Netherlands.
3. Pronk, C. N. A., N. Koopman, and D. de Hoop. 1992. Development Concept for Dutch User Support, Paper IAF-92-0711, *Proc 43rd Congress, International Astronautical Federation*, 28 August to 5 September 1992, Washington D.C., USA.
4. van Eenennaam, J. 1991. Capillary Electrophoresis in Space (CEIS): Payload Simulation, *HPCE Case Study Technical Note 1*, Issue 1, Comprimo Consulting Services b.v., NIVR Contract 2001CO, November 1991.
5. Beckman Instruments Inc. 1989. *P/ACE System 2000 Instrument Manual*. Beckman Instruments Nederland bv, Nijverheidsweg 21, 3640 AA Mijdrecht, The Netherlands.
6. Grant, T. J., M. H. Wigmans, F. Eckhard, and J. van Eenennaam. 1992. A Re-Useable Simulation for Space Payloads, *Proc European Simulation Multiconference*, 1-3 June 1992, York, UK, pps 30 to 34.
7. Dorf, R.C. 1989. *Modern Control Systems*, Addison-Wesley, Reading, Mass., USA.
8. Tate, A., J. Hendler, and M. Drummond. 1990. A Review of AI Planning Techniques. In *Readings in Planning*, Allen, J., J. Hendler, and A. Tate, (eds), Morgan Kaufman, San Mateo, California, USA, 26-49.
9. Marca, and McGowen. 1988. *Systems Analysis and Design Technique*, McGraw-Hill, USA.
10. Chen, P. P. 1976. The Entity-Relationship Model: Towards a unified view of data, *ACM TODS*, 1, 9-36.
11. Coad, P., and E. Yourdon. 1991. *Object-Oriented Analysis*, 2nd ed, Yourdon Press Computing Series, Prentice Hall, Englewood Cliffs, New Jersey, USA.
12. Mitchell, T. M. 1982. Generalisation as Search, *Artificial Intelligence Journal*, 18, 2: 203-226.

VOYAGER UPLINK PLANNING IN THE INTERSTELLAR MISSION ERA

Susan H. Linick and Kathryn R. Weld

N94-23870

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109
USA

ABSTRACT

The Voyager Project has entered its last phase of discovery - the Voyager Interstellar Mission (VIM). Because of the reduced scope of the project and a lower budget, new ways had to be developed to program two spacecraft with fewer people, allowing for some sequence development flexibility without additional risk. In the previous cruise era, it took a seven-person sequence team 12 weeks to develop a nominal eight week cruise sequence. Today it takes a three-person team six weeks to develop a 13 week sequence load. This paper will describe in detail the sequencing strategy which reduces the volume and frequency of sequence loads, and the new tools and processes developed which reduce the manual effort required to generate these sequences without adding risk.

KEY WORDS: Uplink Process, Baseline, Overlay, Ground Blocks and Spacecraft Blocks

1. INTRODUCTION

With many successful planetary encounters behind them, the two Voyager spacecraft began a new pilgrimage - the search for the Heliopause, where the sun's bow shock meets the interstellar medium. This last phase of the mission would primarily concentrate on fields, particles and waves experiments, and astronomy observations by the Ultraviolet Spectrometer (the sole inflight UV observatory in its wavelength range). Because these activities are repetitive, less sophisticated, and not as densely packed as encounter activities, the sequence process by which these activities are planned and processed into spacecraft commands could be streamlined. This became a necessity when the decreased VIM budget resulted in reduced Sequence Team staffing. The small team

could only develop one spacecraft load at a time, so the development time had to be greatly reduced. New software tools to aid in efficiency had to be developed. The challenge was to define, build and transition to this new uplink system in about a year's time to meet the lower VIM funding.

2. THE "PRE-VIM" UPLINK PROCESS

2.1 The Process Flow

In preparing for the Neptune Encounter, the Voyager Project chose an uplink process similar to those used for previous encounters. This process was comprised of two distinct sequencing strategies and it allowed for the generation of "tailor-made" sequences. One strategy was to develop nominal cruise sequences, typically eight weeks in duration. The other strategy, a twofold plan, was used to build the Neptune Encounter sequences. The first of the two encounter plans budgeted for a lengthy advanced planning stage where a detailed (to the command level) sequence was designed using the latest ephemeris data. It was then implemented, approved and placed "on the shelf". The second encounter plan allowed for updates to the "on the shelf" products to incorporate newer ephemeris data as well as any additions, deletions and/or modifications to encounter sequence activities just prior to uplink. Listed below are the typical durations for each sequencing strategy:

Cruise Sequence Development - 12 weeks
Encounter Advance Planning - 22 weeks
Encounter Updates - 12 weeks

2.2 Sequence Team Workforce

The pre-VIM Sequence Team included a Team Chief and Deputy, and four subteams which included

support from seven technical areas to allow for parallel sequence development efforts. This workforce could execute the sequence strategies necessary to build two cruise sequence loads (Voyager-1 & Voyager-2), one advanced planning encounter load and one encounter updates load at the same time.

2.3 Sequence Strategy Success

The success of the Neptune Encounter confirmed the fact that the sequencing strategies used in the preparation of the Voyager Neptune Encounter were effective, sound and complete. Unfortunately, these sequencing strategies were highly labor intensive and would not be suitable for the next phase of the Voyager Project's life span.

3. THE NEW VIM UPLINK PROCESS

3.1 New Mission Objectives

The VIM objectives for both Voyager spacecraft are to investigate the interplanetary and interstellar media and to continue gathering data from the Ultraviolet Spectrometer (UVS), the only operational instrument left on the scan platform.

To accomplish these objectives, the Voyager Project approved a plan to have a series of recurring fields, particles and waves calibrations and digital tape recorder playbacks folded into an "on-board" baseline sequence which would continue to operate on both spacecraft indefinitely, without any required ground intervention. This baseline sequence would also accomplish the necessary activities to maintain spacecraft health and antenna earth pointing. The baseline sequence activity would be augmented by "ground transmitted" overlay sequences of 13 week durations. These overlay sequences would nominally be used for UVS observations, and engineering calibrations.

3.2 Budget Concerns

The VIM staffing budget would only allow for a maximum five-person Sequence Team to accomplish the VIM mission objectives. This meant that there would be no staffing for emergencies, but the quality and completeness of the sequence products could not be sacrificed.

The Voyager Project needed an uplink process which would allow this reduced Sequence Team to continu-

ously produce overlay sequences for both spacecraft, while in parallel, be able to generate unplanned sequences such as heliopause encounter sequences, baseline update sequences or anomaly investigation/corrective-action sequences. In order to meet the 13 week duration criteria for an overlay sequence and allow for unplanned work, this VIM Sequence Team would require a six week sequence development schedule to prevent the overlapping of on-going sequence tasks.

3.3 Assumptions

It was known that during VIM, it would be difficult to acquire seventy meter Deep Space Network (DSN) antenna coverage. In order to limit the number of these requests, the baseline sequence design strategy would require clustering activities with similar DSN telecommunication performance needs. Due to advanced knowledge of baseline sequence activity, other projects would be able to correlate their DSN antenna coverage requirements with key Voyager activities to avoid conflict.

The Project required that no additional operations risk be factored into the new uplink process above that accepted prior to VIM. Fortunately, this requirement could be achieved by (1) continuing to use the Voyager simulation tool, COMSIM, to validate every sequence prior to spacecraft transmission, (2) maintaining the same degree of constraint checking as in the pre-VIM era, and (3) using the overlay sequence to capture baseline sequence playback data lost from weather problems or DSN antenna coverage conflicts.

The Project also mandated that new or updated VIM software programs not be targeted for design and execution on JPL's mainframe UNISYS system unless absolutely unavoidable. Dependence on the institutional mainframe needed to be avoided due to budget, operations and maintenance concerns.

3.4 Implementation Strategies

A VIM uplink process robust enough to comply with the VIM objectives, the budget guidelines and project assumptions, would require major changes in both the spacecraft and ground systems.

3.4.1 Spacecraft System Redesign

The Spacecraft Team led the effort to design a set of eleven on-board software routines (spacecraft blocks) which when called for execution, would be expanded

to the command level and sent to the appropriate spacecraft subsystem to perform the intended activity. The Sequence Team engineered the detailed design strategies to build the baseline sequence. Both spacecraft were then reprogrammed to contain (1) all spacecraft blocks, (2) a table of high gain antenna earth pointing information to keep each spacecraft properly pointed for approximately 25 years, (3) the baseline sequence, comprised of a set of repeated activity calls at a specified frequency to designated spacecraft blocks (looping cyclics), and (4) the Backup Mission Load which reduces the complexity of the baseline sequence and allows it to continue if ground commandability is lost.

3.4.2 Uplink Process Redesign

The uplink process is the operation by which science, engineering and navigation requests are translated into spacecraft computer instructions. For details see the process flow diagram shown in *Figure 1*.

There are three phases to developing a sequence:

design, implementation and validation. The design phase is where inputs are integrated into a sequence plan. The implementation phase converts the integrated sequence requests into commands, and the validation phase translates the commands into spacecraft computer instructions and simulates the sequence for soundness.

The uplink process for the overlay sequence begins six weeks prior to the execution of the sequence. The first week begins the design phase, devoted to the detailed planning of the sequence. The preliminary sequence requests (SRs), which come from the Science, Spacecraft and Navigation Teams are gathered. These SRs are evaluated and costed for computer words and manual work effort. They are then integrated. A strawman timeline is built showing the most current DSN antenna coverage and all planned activities. A coordination meeting is held to work out any issues or conflicts.

The SRs are finalized and formally approved at the beginning of the second week, which is the start of

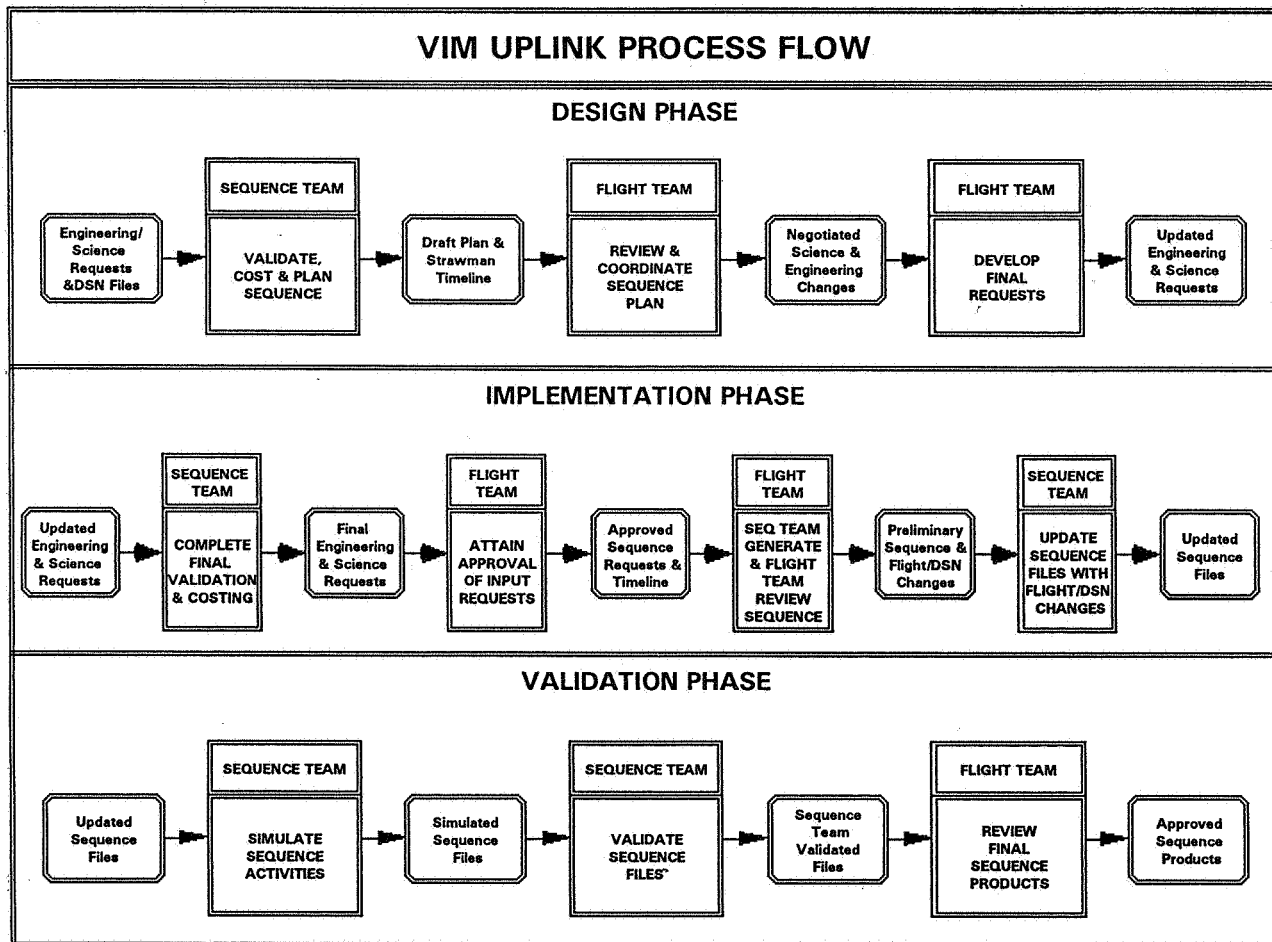


Figure 1: VIM Flow Process Diagram

the implementation phase. The UVS input request file (generated by the Science Team) is merged with hand-generated engineering and navigation requests. Electronic files are generated necessary to build an integrated timeline, the spacecraft command file, and the sequence-of-events document. Review products are generated for a flight team review. Modifications to the sequence are then made based upon review comments and updated DSN antenna coverage. This completes the implementation phase.

The fourth week is devoted to simulating the sequence activities through the Voyager COMSIM program to ensure all spacecraft systems will operate correctly. The end product of this simulation is the Ground Command File (GCMD). Validation follows simulation and, by the fifth week, review products are generated for a final review by the flight team. This completes the validation phase.

During the sixth week, the GCMD is translated into a spacecraft command file and uplink approval is attained. The load is then transmitted to the spacecraft by the DSN. The Monday following this uplink marks the execution start of the new sequence, and the beginning of the uplink process for the other spacecraft.

The process to modify the baseline sequence follows the same operational scenario as the overlay, but is drawn out over a period of four months to lessen its impact on the concurrent, more highly time-constrained overlay sequence development. The baseline sequence was developed using looping cyclics of spacecraft blocks, which are automatically restarted each year. It is at this point that modifications can be introduced, allowing for change only once per year.

The capability to generate unplanned sequences (commands sent to the spacecraft in a time-critical manner) is a necessary part of operations. These sequences are not scheduled in the normal way. They must be built quickly with careful correlation to the ongoing spacecraft activities. The development milestones for these sequences are conducted in parallel with scheduled sequence tasks.

The new process allows the Sequence Team to operate within the maximum five-person limit. A three-person rotating sub-team is used for nominal overlay sequence development. The remaining personnel are involved in baseline and unplanned sequence development work, software upgrade user and regression testing, and ground block database

maintenance. The rotation of duties on a regular basis keeps the knowledge current and minimizes turnover impact.

For two years the VIM uplink process has proven its merit in the development of the baseline, overlay and unplanned sequences. Since that time, both spacecraft have been performing with greater than expected science and engineering return.

3.4.3 Ground System Redesign

The software system supporting the uplink process is typically called the Mission Sequence System (MSS). A constraint on the redesign of the VIM MSS was that new development be PC based since the Sequence Team was already equipped with compatible machines.

The major redesign efforts within the MSS were concentrated in altering or building new programs for the design and implementation phases of the new six week sequence development process. The pre-VIM process used eight weeks for these two phases and was highly labor intensive. The goal was to not only automate tasks as much as possible, but to also replace entire sequence team positions with software capabilities. Today, three weeks are needed to complete the design and implementation phases.

Just as the spacecraft system redesign relied heavily upon the use of spacecraft blocks, the MSS redesign required the use of established ground blocks to define requested overlay sequence activity. The ground blocks were designed to group relatively timed commands for frequently used sequence activity. This allows the user to furnish a minimum set of parameters to initiate the block implementation. These ground blocks not only documented the necessary command activity for a specific sequence request, but also included the detailed notation needed to generate human readable sequence review products.

Figure 2 represents the data flow for the redesigned MSS using the associated acronyms mentioned below.

The Design Phase of the VIM uplink process was enhanced by developing four programs to accomplish the following:

- Build UVS input request files for overlay activities (UVSPRC).
- Build a working strawman timeline containing

baseline & UVS overlay activities, DSN view periods, antenna coverage and associated data rate capabilities (Strawman T/L).

- Constraint check for data rate capabilities (TSP).
- Constraint check for scan platform and UVS pointing violations (AACSPRC).

The Implementation Phase was upgraded by modifying one existing program and designing two new programs to achieve the following:

- Merge Science, Engineering and Navigation Requests and expand to the command level (BLKPRC, AACSPRC).
- Check for ground and spacecraft constraint violations (BLKPRC, TSP).
- Translate the overlay sequence into binary bits readable by the on-board computer (SEQTRAN).
- Build the integrated baseline & overlay sequence timeline (Laserjet T/L).

The Validation Phase, where COMSIM is executed, was updated to allow for the following:

- Enable COMSIM to simulate the modification made to the spacecraft flight software.
- Pass ground block notation to the Sequence of Events (SOE) software.

3.4.5 Ground System Redesign Costs

The Mission Sequence System Functional Requirements Document (MSS FRD) was completed in a four month time span. This was due to the fact that experienced users worked side by side with software engineers to coordinate necessary software capabilities with VIM operational scenarios. Existing flight team members participated in this effort and no additional staffing was required to complete the MSS FRD.

Software implementation of the MSS redesign was folded into a two phase development plan. The Phase 1 delivery contained the minimum set of programs needed to generate command files for baseline and overlay sequence transmission. The Phase 2 delivery completed the set of programs needed to allow the VIM Flight Team to generate an overlay sequence within the allotted six week sequence development schedule. Phase 1 was completed in six months with Phase 2 following six months later. The existing software development staff of three was augmented by one additional programmer for a six month period.

Before using the new MSS to generate real sequenc-

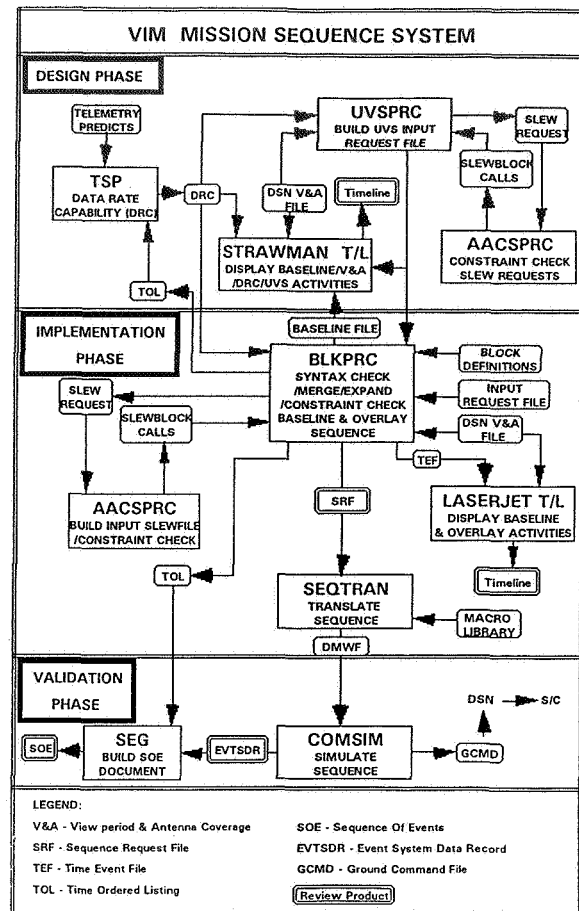


Figure 2: MSS Data Flow

es, an extensive User Test Plan was developed and executed. The plan included the involvement of appropriate flight team members to review sequence products containing every possible combination of ground block calls. The plan also required the simulation of all ground blocks by COMSIM to insure confidence in their usage. The Sequence Team added three personnel for an approximate six month period to complete the user testing.

One year and two months after the Voyager Neptune Encounter, the Phase 1 MSS was used to build the first overlay sequence for spacecraft transmission. Six months later, the Phase 2 MSS was delivered and the six week sequence development schedule was realized.

4. MODIFICATION EFFECTS

The use of blocks to define standardized activities promotes ease in operational sequence implementation (for the requester and the implementer) with main-

tained reliability. Operating with blocks limits flexibility and science optimization, but allows for automation in uplink development and increases team productivity. Using pre-defined activity blocks also makes training new personnel easier.

Placing a large volume of blocks on-board the spacecraft decreases computer sequence memory, but allows the baseline sequence to operate each spacecraft autonomously.

Transition to the VIM uplink process necessitated the following changes: (1) new procedures had to be written and old ones revised or deleted, (2) fewer sequence products were generated and fewer review meetings needed, (3) the nominal sequence development schedule was shortened from 12 weeks to 6 weeks, (4) greater process automation reduced errors, (5) staffing was reduced thereby saving dollars and (6) the Science, Spacecraft and Navigation teams now play a more integral part in designing each overlay sequence.

5. FUTURE APPLICATION

The VIM uplink process is simple, stable and easily adaptable to the new Discovery class missions of the future. The process is usable for a standalone mission or can be considered adaptable for shared project operations. The Voyager project endorses this new idea and is open to the challenge of a shared project operations environment.

The process can support a planetary cruise effort with few modifications to existing operations plans and procedures. What the process does lack are the necessary steps to incorporate instrument pointing with correlated maneuver strategies. Since these steps were there before VIM, there would be little problem to reinstate them. What makes the VIM process appealing is the baseline/overlay sequence concept, the established interteam/intrateam interfaces and the proven set of procedures for flight team operations.

The ability to accommodate an encounter sequence strategy was built into the VIM uplink process. This part of the process has already been tested by the flight team in the last few weeks. Sequence operations generated special sequences to investigate unexplained radio emissions recently observed by the waves experiment.

6. CONCLUSIONS

The entire Voyager uplink process was transformed to meet the scaled down requirements of the interstellar mission. The many changes described in this paper (using on-board spacecraft blocks and ground blocks, building baseline/overlay sequences, using new PC driven software to simplify the process and modifying the process itself) all resulted in a myriad of savings of time, level of effort, and dollars. The process now fits the task.

The success of VIM's operational redesign is largely due to the fact that the user community was allowed to design a new system using their own experience and creativity.

The primary reason the Voyager Project has been able to automate many of its operations is due to reduced mission complexity. VIM has become an example of a "cheaper, faster, better" mission that the new NASA philosophy holds as the wave of the future. This type of mission is only possible if the idea of "simple" is included in every aspect of the project.

Voyager has been at the forefront of solar system exploration and is now leading the way towards establishing a mission operations concept applicable for Discovery class missions.

7. ACKNOWLEDGEMENTS

The authors wish to express their thanks to the many creative individuals on the transition team for their hard work in making all components of the VIM uplink process a reality. Specifically, thanks go to Marie Deutsch and Lanny Miller for their excellent systems engineering support, George Masters and Sheldon Herman for their creative design strategies, and Donald Heller, Brian Ross, Richard Lemon, Lawrence Zottarelli, Judith Morris and Carlos Carrión for their dedicated involvement in augmenting the effort through completion.

The work described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

Randii R. Wessen and Daniel F. Finnerty

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California, U.S.A.

N94-23871

ABSTRACT

This paper will address the science planning and sequencing aspects of the command generation process for the scientifically diverse Cassini Mission. The mission's prime objectives are to study the Saturnian system and deliver the Huygens Probe into its major moon Titan. Together, the spacecraft and probe will be the largest and most complicated craft ever launched to another planet.

The presentation will begin with an overview of the Cassini spacecraft and its scientific instrumentation. This will be followed with a description of the October 1997 mission. Next, the structure of the science planning and sequencing process, with special emphasis on science's role, will be outlined. Finally, this presentation will conclude with a discussion of some of the unique challenges faced by the Ground System during Cassini's four-year orbital tour.

Key Words: Astronautics, planetary exploration, operations, science planning, Saturn

1. Cassini Spacecraft

The Cassini spacecraft is a 3-axis stabilized craft with a design lifetime of 12 years. It is divided into four core elements. The first element is a rigid four meter diameter High Gain Antenna (HGA) for reception of commands from Earth, transmission of science and engineering data from the spacecraft, relay antenna for the probe data and the source of the output RADAR signal. The HGA is attached to the Upper Equipment Module (UEM).

The UEM is the second core element. This element is a 12-sided bus which provides storage for the major electronic components (e.g., Command and Data Subsystem (CDS), Attitude and Articulation Control Subsystem (AACS), Solid State Recorders (SSRs), etc.). It also provides the mounting structure for the 11 meter long Magnetometer Boom, one articulating reaction wheel, the Remote Science Pallet, the Fields & Particle Pallet and their associated orbiter science instruments.

The third element, located under the UEM, is the Propulsion Module (PM). This module contains all the bi-propellant fuel for the mission and will be used for all major trajectory correction maneuvers. The Cassini fuel load consumes a significant amount of spacecraft mass. For comparison, the Voyager

spacecraft had a total mass of 825 kg (100 kg of mono-propellant). The Galileo spacecraft had a launch mass of 2561 kg (959 kg of bi-propellant). Cassini and the Huygens Probe will have a total mass of 5120 kg (3000 kg of bi-propellant, approx. 60%).

The last major core element is the Lower Equipment Module (LEM) and is located underneath the PM. This module houses the two 490 N main engines, the four thruster clusters for attitude control, three reaction wheels for non-propulsive maneuvers and the three Radioisotope Thermoelectric Generators (RTGs) which together at launch will produce approximately 850 watts of electrical power.

1.1 Scientific Investigation

The Cassini science instruments were selected based on their ability to increase our knowledge of the Saturnian system by at least an order of magnitude in their respective areas. Each investigation performs unique science by itself and complements the other investigations by performing many synergistic observations.

The instruments can be divided into three classes; Passive Remote Sensing instruments which must be pointed at a specific target-body and then processes the externally generated source signals; Active Remote Sensing instruments also must be pointed at a specific target-body to collect data but in this case, the source signal processed is generated onboard the spacecraft; and finally the Fields, Particles & Waves instruments which measure the in situ environment.

1.1.1 Passive Remote Sensing Investigations

There are four Passive Remote Sensing investigations. All of them are co-aligned with the narrow angle camera boresight and mounted onto the Remote Sensing Pallet (located on the UEM). Alphabetically, the first investigation is the Composite Infrared Spectrometer (CIRS) which has two linear 1×10 arrays. The total field of view (FOV) of each array is 2.9×0.3 mradian. The arrays are separated from each other by 0.67 mradians. The remaining CIRS FOV is circular and has a diameter of 4.3 mradians. With these FOVs, this instrument covers the infrared spectrum from 10 to 1000 micrometers. Its primary objective is to generate spectral maps to study temperature and composition of Saturn and Titan's atmospheres, satellite surfaces and ring particles.

Next is the Imaging Science Subsystem (ISS). This subsystem is composed of two CCD cameras. The wide angle camera has a 61.2 mradian square FOV with a spectral response from 0.38 to 1.1 micrometers and the narrow angle camera has a smaller FOV of 6.1 mradian square but a slightly wider and "bluer" spectral coverage. Its response ranges from 0.2 to 1.1 micrometers. ISS's primary objective is to obtain multi-spectral images of Saturn, its satellites and rings.

The third Passive Remote Sensing instrument is the Ultraviolet Imaging Spectrometer (UVIS). It has two rectangular FOVs. The narrow FOV is 0.5 x 64.2 mradian while the wider FOV is 6.42 x 64.2 mradian. The instrument's spectral coverage ranges from 0.055 to 0.19 micrometers. The investigation's major objectives are to obtain spectra and low resolution imaging to determine structure, chemistry and composition of Saturn's atmosphere and rings.

The last Passive Remote Sensing instrument is the Visual and Infrared Mapping Spectrometer (VIMS). This investigation also has two FOV. The smaller infrared FOV is 32 mradian square and covers a spectral range from 0.95 to 5.2 micrometers. The visual channel has a 42.7 mradian square FOV and covers 0.35 to 1.05 micrometers. VIM's prime objective is to study the composition and surface structure of Saturn's atmosphere, rings and satellite surfaces.

1.1.2 Active Remote Sensing Investigations

The Cassini payload has two Active Remote Sensing investigations. The first is RADAR. This instrument uses the HGA to transmit and receive a 13.8 GHz Ku-Band signal at a specific target-body. Its primary objective is to perform radar imaging, altimetry and radiometry observations of Titan's surface.

The other investigation is the Radio Science Subsystem (RSS). This investigation uses the HGA to transmit a signal through a medium (e.g. interplanetary space, rings, atmospheres) to be received by Earth-based receiving stations. In this case 2.3 GHz S-band, 7.18 to 8.42 GHz X-band and 32 to 34.3 GHz Ka-band frequencies are used. The investigation's primary objectives are to study Saturn's atmosphere, ionosphere, ring system and its gravitational field. In addition, the gravitational fields associated with the icy satellite, Titan's atmosphere and ionosphere will be studied. During cruise, a search for gravitational waves will be attempted.

1.1.3 Fields, Particles & Wave Instruments

There are six Fields, Particles & Waves (FP&W) instruments on the Cassini spacecraft. These investigations usually require some type of internal articulation or spacecraft rotation to obtain spatial information about the distribution of the phenomena being measured.

The Cassini Plasma Spectrometer's (CAPS) primary objective is to study in situ the plasma environment within and near Saturn's magnetic field. The investigation is composed of three detectors integrated together and mounted on the Fields & Particle Pallet. The three detectors are the Electron Spectrometer (ELS), the Ion Mass Spectrometer (IMS) and the Ion Beam Spectrometer (IBS). The combined system will articulate back and forth up to 200 degrees in a "windshield wiper" type of motion.

Next is the Cosmic Dust Analyzer (CDA). It consists of an aperture that has a 100 degree FOV and is sensitive to particle impacts, masses, speeds and charges. The device is mounted directly to the UEM and can be articulated back and forth up to 270 degrees. Its primary objective is to study in situ ice and dust grains in the Saturnian system and in interplanetary space.

The third FP&W investigation is the Ion and Neutral Mass Spectrometer (INMS) and is mounted to the Fields & Particle Pallet. Its prime objective is to study in situ the composition and concentration of charged particles and neutral gases, primarily near Titan.

The Dual Technique Magnetometer (MAG) uses both a Vector/Scalar Helium Magnetometer (VHM) and a Fluxgate Magnetometer (FGM) to study Saturn's magnetic field and its interaction with the solar wind. These instruments are located on a deployable 11 meter Galileo-type MAG Boom. The FGM is located somewhere past the boom halfway point while the more sensitive VHM is located at the very end.

The fifth FP&W investigation is the Magnetospheric Imaging Instrument (MIMI). This investigation is divided into three separate devices, the Charge-Energy-Mass Spectrometer (CHEMS), the Ion and Neutral Camera (INCA) and the Low Energy Magnetospheric Measurement System (LEMMS). Only CHEMS and LEMMS are mounted on the Fields & Particle Pallet. INCA is mounted directly to the UEM. LEMMS has the capability to be articulated to gain spatial information on time scales shorter than permitted by the spacecraft turn rates. MIMI's primary objectives are to obtain global magnetospheric imaging and in situ measurements of Saturn's magnetosphere and solar wind interactions.

Finally, the Radio & Plasma Wave Science (RPWS) investigation is composed of three deployable dipolar antennas, a cluster of Magnetic Search Coils and a deployable Langmuir Probe. All components of this investigation are mounted to the UEM. The primary objective of this investigation is to study plasma waves, radio emissions and dust in the Saturn system.

1.1.4 Huygens Titan Probe Instruments

The 2.7 meter Huygens Titan Probe is being built by the European Space Agency. It has a mass of approximately 300 kg and houses six scientific investigations. They consist of the Aerosol Collector and Pyrolyser (ACP), the Descent Imager and Spectral Radiometer (DISR), the Doppler Wind Experiment (DWE), the Gas Chromatograph and Mass Spectrometer (GCMS), the Huygens Atmospheric Structure Instrument (HASI) and the Surface Science Package (SSP). Together this collection of instruments, combined with Cassini's remote sensing instruments, will make the most comprehensive study of Titan's atmosphere and surface to date.

2. VVEJGA Mission

The sheer mass of the Cassini spacecraft precluded the use of a direct trajectory to Saturn. In order to obtain sufficient energy, a multiple gravity assist trajectory had to be selected. The Cassini trajectory, known as Venus-Venus-Earth-Jupiter Gravity Assist mission (VVEJGA), requires almost seven years of cruise to reach Saturn. The impacts of this trajectory are a more stressful thermal environment for the spacecraft, less power at end-of-mission due to the degradation of the RTGs, a longer cruise and more complicated ground operations.

2.1 Interplanetary Cruise

The launch period for the Cassini mission opens on 1997 October 6. The current plans call for a launch with a Titan IV/Centaur with strap-on Solid Rocket Motors Upgrades (SRMU). However, the mission has maintained the flexibility to use the lower performance Solid Rocket Motors (SRMs) should the SRMUs not be available for launch.

Upon liftoff from the Kennedy Space Center in Florida, Cassini will have to perform the first of its many maneuvers 25 days after launch. During the initial cruise portion of the mission, there will be minimum instrument calibration, characterization and maintenance. Cruise will consist primarily of engineering and navigation activities. There are no plans at this time to obtain science during the

interplanetary cruise or at any planetary encounter prior to two years before Saturn orbit insertion.

The Venus1 swingby will occur six months after launch and will bring Cassini within 300 km of the planet on 1998 April 21. The swingby past Venus will put Cassini on a 14 month orbit, taking the craft out past the orbit of Earth and then back in for its second rendezvous with Venus.

Between Venus encounters, at Launch+14 (L+14) months, the majority of the instrument checkouts will be performed. At L+18 months, limited instrument checkouts will be completed and routine simple instrument maintenance will begin. Two months later at L+20 months, on 1999 June 20, the Venus2 swingby will occur.

The second swingby of Venus occurs when the Earth and Venus are close to each other in their respective orbits. Consequently, the Earth swingby occurs only two months after the Venus2 swingby (1999 August 16). The close placement of these encounters will drive ground operations in order to prepare the required maneuvers prior to and following each encounter. Earth avoidance after Venus2 is another prime concern and which will complicate the maneuver strategy. In total, there will be 15 spacecraft maneuvers between launch and L+30 months.

After the Earth swingby, Cassini will head out of the inner solar system. The last gravitational assist will be with Jupiter a little more than three years after launch (2000 December 30). The aim point at Jupiter will be more than 20 million km (>140 Jupiter radii) from the planet. Cassini will arrive at Saturn on 2004 June 25.

2.2. Orbital Operations

The current tour is known as 92-01, has 63 orbits around Saturn and though not approved by the investigators, is typical of the type of science opportunities possible at Saturn.

Cassini will approach Saturn's south pole from the dayside prior to beginning its four-year tour of the system. The initial approach will allow some unique ring science observations as the craft approaches the planet at a relatively high inclination. The craft will climb above Saturn's ring plane at which point the Saturn Orbit Insertion (SOI) maneuver will occur. This maneuver will last approximately 70 minutes and will commence at periapsis.

The SOI maneuver will place the craft in a 152 day orbit about Saturn. This first orbit, referred to as petal 0, sets up the correct geometry for the delivery

of the Huygens Titan Probe. Approximately 134 days into petal 0, the Huygens Titan Probe will be released from the Cassini spacecraft. For the next 21 days the probe will continue by itself on a ballistic trajectory into the atmosphere of Titan. Upon atmospheric entry, timers on the probe will activate the six science instruments, release the heat shield and the main parachute.

During the Descent Phase, which lasts approximately 2.5 hours, the Huygens Titan Probe will relay atmospheric composition, temperature and pressure information back to the Cassini spacecraft. In order to acquire the data, the craft will be turned such that the HGA acts as the relay antenna for the probe. The information will be redundantly stored on board the SSRs to be played back at a future date.

After the probe mission, the early orbits are designed to precess in a counter-clockwise fashion (as seen from Saturn's north pole) in order to obtain low solar phase angle coverage of the planet. After petal 10, the orbits will be changed to precess in a clockwise direction. This precession continues till late in the mission when the orbit periods are shortened in order to increase the petal inclination. By the end of the mission, an inclination of greater than 75 degrees will have been achieved.

Over the course of the four year mission, Cassini will perform 33 Titan swingbys. The altitudes of the Titan swingbys will range from 950 km to greater than 5000 km above its surface. Almost a third of the Titan passes will be at an altitude of 950 km. There also will be opportunities for 10 Earth- and 10 Sun-Titan occultations.

There also will be four targeted and 29 non-targeted satellite encounters. The targeted satellite encounters will be of Enceladus, Dione, Rhea and Iapetus, and will occur at an altitude of 1000 km or less. The non-targeted encounters are all under 100,000 km. Voyager 1 and 2 combined had only four Saturnian satellite encounters less than 100,000 km.

The beginning of the tour also provides for five Earth- and Sun-Saturn/ring occultations. Near the end, when the spacecraft inclination increases, the number of occultations will be 27 Earth- and 9 Sun-Saturn/ring occultations.

3. Science Planning & Sequencing

The science planning and sequencing aspects for the Cassini mission will be based on a fundamentally different approach from its predecessors Voyager and Galileo. The concept is analogous to the one used for Mars Observer and is based on distributed science

operations. This will allow the investigators to perform their science operations remotely at their home institutions via networked connections to the Project information system.

This differs from past missions where investigators had to periodically travel to JPL to be apprised of the sequence development status. During cruise portions of a mission, these meetings might have happened two or three times a year. The level of travel increased to the point that during planetary encounters, investigators were resident at the laboratory for weeks and even months at a time.

A distributed science operation approach should mitigate the need to transfer to or duplicate instrument expertise from the home institution at JPL. Investigators will be responsible for identifying science opportunities, planning science observations and generating instrument commands.

3.1 Mission Planning

The Ground System (GS) will be responsible for mission success at the system level. As such, mission planning will involve defining a physical environment that allows each investigation to operate successfully acquiring its desired data.

In order to accomplish this critical task, mission planners, working with the investigators, spacecraft engineers and navigators, will compile information pertaining to science objectives from which a desired trajectory can be developed. This trajectory must also take into account the conditions in interplanetary space and during orbital operations. As an example, spacecraft thermal considerations and instrument performance characteristics in radiation environments will be evaluated and incorporated into the trajectory selection.

Along with the definition of a trajectory, high priority science observations, especially those tied to geometric events, will be identified such that spacecraft resources (e.g. instrument power, data volume, pointing, etc.) can be allocated to meet the objectives of the investigators and the mission.

To accomplish this difficult task, mission planners will have to work very closely with the remotely located investigators.

3.2 Science Planning

Science planning will require an intensive amount of work identifying science opportunities, critical observations and negotiating spacecraft resources.

The Science Planning and Operations Team (SPOT), which is located at JPL, first produces a working timeline from mission plans. This timeline will contain all major spacecraft maneuvers, maintenance activities, operational modes, navigation requests and Deep Space Network (DSN) passes. The working timeline can best be thought of as the backbone on which the science observations will be placed.

Once generated, the working timeline, combined with resource envelopes/allocations, will be placed in the Project Data Base (PDB) to allow the investigators access to the information. At this point the investigators will define observation opportunities, optimal observing sequences (within allocated resources) and note which engineering activities need to be shifted (within their stated tolerances) to resolve initial observation conflicts.

It is envisioned that a number of software tools will be available to help the investigators identify observation opportunities. One such tool planned for development at JPL is called SOA (Science Opportunity Analyzer). This program will incorporate (user-definable) planetary system physical models to allow the user to search for specific observing conditions and to characterize observing conditions at a particular time and location. The program will also be able to produce animated displays of planetary system targets as seen from a user specified vantage point.

Upon completion, the updated information in the PDB will be accessed by SPOT; resource utilization, observation conflicts and constraint violations will be identified. Once determined, the Science Operation (SO) Teams (Principal Investigators and Facility Engineering Operation Teams) will resolve conflicts, specify instrument states and generate specific activities which will contain macro-level instrument parameters.

It will be critical for SO Teams to confirm that their observations do indeed fit into their allocated resources. This will be accomplished by using instrument models for all managed resources, such as power. Any observations found, by system-level checks at JPL, to violate a specific resource envelope will be subject to removal from the sequence. The GS will only be responsible for checking system-level commands for compatibility with system-level resources and constraints.

The SO Teams will then place their detailed instrument observation definitions into the PDB as instrument-specific Observation Defining Records. These records are accessed by SPOT to generate an

integrated Activity Plan (AP). The AP will be science's input into the sequence generation process.

3.3 Sequence Generation

The sequence generation process uses the Activity Plan to produce engineering and facility instrument sequence inputs. These inputs are expressed as spacecraft activities and parameters and are combined with the required ground events. From these inputs, an integrated sequence is generated using the Mission Sequence System (MSS). The MSS generates the sequence and performs a majority of the system level constraint checks. All MSS output products are reviewed for validity by the operation teams. In addition, a high-speed spacecraft simulator will be used to confirm the operation team's conclusions. The final output product is the Ground Command (GCMD) File, and once approved by the Project, is transmitted to the spacecraft.

A limited number of changes to the GCMD may be possible prior to sequence execution. These modifications may be needed to account for changes in the Saturnian system for which no apriori knowledge of the event was known.

4. Challenges

As with any new approach, many challenges lay ahead during the GS's implementation. For Cassini, the situation is exaggerated due to the deferred development of major portions of its ground system until post-launch.

Of the many challenges faced by the Cassini GS, the approach for resolution of conflicting activities will be one of the more difficult to solve. In most of the past planetary missions, sequence conflicts were resolved in a centralized location where all participants could argue "head-to-head" until a solution was agreed upon.

For Cassini, "head-to-head" meetings will not be practical during the four years of orbital operations. Other than for SOI observations and Huygens Titan Probe data, no one science observation is so critical that a compromise can not be found. It will be necessary to resolve sequence conflicts remotely at the investigator's home institution.

One approach for doing this is the early identification of high priority science opportunities and critical engineering activities. Once determined, the JPL-based mission planning function can assign initial operational modes, power envelopes, primary targets and associated data volumes. Once these envelopes have been assigned, the resulting profiles can be

placed in the Project Database for remote access by investigators.

It is important to note that this approach does not eliminate all sequence conflicts; it only reduces the number of conflicts between high and low priority activities. However, this will reduce the number of iterations the sequence goes through during its development.

The savings come from the very nature of the sequence development process. This process involves proceeding from lower to a higher level of detail. Conflict resolution often results in a need to return to a lower level of detail to fix the problem, rendering much detailed information obsolete. The early identification of high priority activities, and timely elimination of conflicts between them, will be needed to help resolve the sequencing issues in the short period of time available for sequence development.

Following are some of the other obvious GS challenges that have to be addressed.

4.1 No Articulating Instrument Platforms

Cassini was initially designed with a high precision pointing platform for the Passive Remote Sensing instruments and a spinning Turntable for FP&W investigations. The removal of these platforms simplified the design and reduced the hardware cost of the spacecraft.

However, one effect of these changes is that during orbital operations, the spacecraft will spend up to 16 hours per day maneuvering off Earth-point to acquire Remote Sensing data. The remaining eight hours will be spent on Earth-point, replaying the Remote Sensing data and collecting FP&W data as the craft slowly rolls about the HGA axis.

The division between the collection of FP&W and remote sensing data, coupled with the potentially greater risk of spending up to 16 hours per day out of contact with the Earth, will add complexity to the spacecraft's flight software and ground operations.

4.2 Power Limitations

Cassini's electrical power output is insufficient to power all the instruments simultaneously during orbital operations. Thus, operational modes have been defined in which each investigation has been allocated a power level for the duration of the mode. The current operational modes are: Optical Remote Sensing, Downlink FP&W (i.e., simultaneous with SSR playback), FP&WI (I for INMS on), RADAR and RSS.

Each SO Team will be responsible for ensuring that their investigation does not exceed its assigned power envelope. The GS will not be able to model instrument power levels commanded by the investigators. However, resource envelopes will account for the expected ranges of power utilized by each instrument during a given mode. System-level checks will be performed to ensure that power levels do not exceed spacecraft capability, triggering a system fault protection algorithm.

4.3 Distributed Science Teams

The concept of remote science operations based on distributed science teams is modeled on the structure of the GS for Mars Observer (MO). At this point in time, MO has just begun cruise operations and is about 10 months away from orbital operations. It may require several months of orbital operations before the full advantages and possible weaknesses of this concept become apparent. Cassini will be eager to incorporate lessons learned from MO into its GS Design.

In MO's case, the instruments are independent of each other, have fixed resource envelopes and the mission is defined as a systematic mapping mission. Designing the remote science operations concept for the Cassini mission, with highly interdependent science observations, tight spacecraft resources and a multi-target mission, will be a challenge.

5. References

1. Cassini Project Policies & Requirements Document; JPL Internal Document; PD 699-004; 1992 July.
2. Cassini Mission and Spacecraft System Delta Preliminary Design Review; JPL Internal Document; 1992 August 11-13.
3. Cassini Ground System Requirements Review; JPL Internal Document; 1992 September 30.
4. R. E. Diehl & J. C. Smith; "92-01 Satellite Tour Design"; Cassini PSG; JPL Internal Document; 1992 October 6.
5. Cassini Ground System Functional Requirements Document; Rev.A; JPL Internal Document; PD 699-500; 1992 September.

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

TECHNIQUES AND SOFTWARE FOR OPTIMUM AND EFFICIENT MISSION SCIENCE SEQUENCE DEVELOPMENT

David A. Bliss
Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California 91109

N94-23872

ABSTRACT

Highly successful mission operations requires efficient and cost-effective science sequence development. Of key importance is the Science Planning and Operations Team's (SPOT's) ability to complete science observation design and integration early in the sequence development process (i.e., before the sequence enters into a formal change control process). Once under formal change control, careful change paper documentation, Flight Team checks, and mission software checks make sequence changes more labor-intensive. This paper discusses team organization, strategies, scheduling, and software employed by the Voyager and Galileo SPOTs to complete science observation design and integration early in the sequence development process.

Key Words: Aerospace, mission operations, science sequencing

1. INTRODUCTION

The schedule for science observation design and integration can vary somewhat from Project to Project. Traditionally, the responsibility for the preliminary phase of mission sequence development belongs to either the mission's Science Planning and Operations Team (SPOT) or to the SPOT in combination with a sequence integration team. During the early sequence development phases, observations are typically designed and integrated to a specified level of detail. During later sequence development phases (i.e., after sequencing changes are under Project Change Control) the final observation design parameters are implemented and detailed mission constraint checking software is run. Experience obtained from both Voyager and Galileo missions suggests that, to maximize science return while minimizing mission operations costs and Flight Team operations impacts, it is best to complete the science observation designs (with the exception of specified param-

eters such as imaging filters and exposures) and integration while the sequence is still in its preliminary phase. By consulting with the spacecraft engineering team, the SPOT would assign timeline windows and resources for engineering events to be incorporated in detail during final sequence integration. Such early completion of the science observation designs provide the following benefits:

- a) Eliminates or greatly reduces the amount of change-paper which must be generated to incorporate changes during final sequence integration.
- b) Avoids the extensive inter-team constraint checks and reviews which changes must undergo during final sequence integration.
- c) Minimizes sequence reintegrations and software reruns from changes which impact the position and timing of other science and engineering events.
- d) Allows the SPOT to focus its attention and efforts on the development of sequences to exploit science opportunities in other mission phases.

By limiting working team size and by utilizing appropriate software tools, the SPOT can achieve early science sequence development.

2. EARLY SCIENCE PLANNING OR SEQUENCE SCOPING PHASE

Before explaining the advantages of early science sequence development, let's first establish an understanding of the overall sequence development process.

2.1 Overview of Sequence Development

The process begins with the identification and selection of the science objectives. Science objectives selection is achieved through a number of meetings with the planetary or science discipline working groups, composed of

investigators from the project experiment teams, sifting through and prioritizing an exhaustive list of possible mission science objectives and resolving inter-experiment conflicts. Following publication of the results of these meetings, the SPOT uses orbit and/or trajectory information from the Navigation Team to compose a science planning guide for the mission. This planning guide establishes prioritized objectives for each mission phase (for Galileo, a mission phase is an orbit composed of several spacecraft command loads) and identifies observation periods which obtain as many science priorities as possible within Project capabilities or spacecraft resources. The science discipline working groups then review and make final planning guide modifications.

After completion of the science planning guide, the formal process of sequence development begins. The first step in sequence development can generically be called Science Scoping (SS). During SS, a science integrator works with instrument experiment representatives (ERs) to assign observation windows and assemble an observation timeline and time ordered listing (TOL) for a specific mission phase. Resources are allocated to each observation and preliminary observation designs are formulated. Timeline windows are identified for spacecraft engineering activities and resources are assigned to these windows.

The second sequence development step can generically be called the Sequence Plan (SP). During SP, the science integrator works with experiment representatives to develop and integrate observations in greater detail. The ERs use mission design software to design observations. Observation parameters which claim spacecraft resources (i.e., the scan platform, telecom, spacecraft power, tape recorder, onboard computer memory, etc.) are input and run through mission guideline and constraints checking software. An updated timeline and time-ordered listing is generated with liens to be resolved in the subsequent Sequence Integration step.

The third sequence development step can generically be called Sequence Integration (SI). During SI, the sequence integration team receives detailed inputs (i.e., scan platform slewing targets, mosaic sizes and scanning rates, instrument configuration details, etc.) from the SPOT and spacecraft engineering teams. The sequence integration team then runs the sequence through engineering activity and observation design parameter check-

ing software and produces an updated timeline and time-ordered listing. Starting with SI, all sequence fixes are under Project Change Control (PCC) and hence must be documented. Minor sequence fixes, such as instrument parameter changes, normally require fairly simple formalized change-paper which are received and acted on only by the sequence team. More involved fixes, such as observation redesigns or timing changes, generally require more extensive change-paper documentation and must receive inter-team review and project approval.

The forth and final sequence development step can be called the Sequence Command Generation (SCG). During the SCG, the sequence is run through spacecraft command and simulation software and spacecraft readable commands are generated. Science changes at SCG are generally allowed only to make a high priority observation work or to preserve an instrument's health. All changes made at SCG require extensive change-paper documentation and must receive inter-team review and project approval.

To enhance sequence development efficiency, this paper proposes and discusses ways to achieve complete science observation development and integration by the end of the Sequence Plan phase (i.e., before sequence changes become subject to PCC).

3. IMPROVED SCIENCE SEQUENCE DEVELOPMENT PROCESS

On the Galileo Project, we have achieved early science observation design and integration by adjusting the Science Scoping and Sequence Plan steps as discussed below.

3.1 Science Scoping

Science Scoping is divided into three stages: development, review, and update. During the development stage, the science planning and operations team develops a formal Science Scoping product for investigator review. The review stage is simply a meeting in which the investigators review the Science Scoping product. The update stage involves incorporating investigator comments into the science scoping product. We have streamlined and modified the development stage (which comprises approximately 75 percent of the Science Scoping step) as described below to achieve an integrated Science Scoping product which is based on and includes detailed observation designs.

To start the Science Scoping development stage, the science integrator holds a kickoff meeting in which he or she proposes the guidelines and schedule for the mission phase sequence development, identifies key information (i.e., telecom and engineering constraints) pertinent to the sequence development, reviews the mission phase science objectives from the science planning guide, and identifies major issues for the sequence development. The ERs, along with the science integrator and technical support personnel, comprise the working team which develop the Science Scoping product.

A key element of the science integrator's schedule is the frequency and duration of the working team integration meetings. During these meetings, the science integrator works with the ERs to formulate observation design strategies and to develop an observation timeline. Given Galileo's eight week Science Scoping duration for each mission phase sequence, we have found that two integration meetings per week, each of approximately two hours duration, is sufficient to resolve integration issues while allowing a maximum amount of time for ERs to design observations.

Since it's important that the integration meeting discussions are focused and efficient, we find that it is best to limit working team size to one experiment representative from each instrument team. Teams whose observations do not require extensive scan platform design work, such as fields and particles teams, may choose to have one ER represent several instrument teams.

In the first integration meeting, the science integrator may allocate spacecraft resources and propose a timeline of observations to accomplish the mission phase's prioritized science objectives. The timeline also includes and assigns resources to windows which will be filled by important engineering activities during the SP and SI steps. Using this timeline as a starting point, the ERs propose modifications based on their instrument team's desires. The science integrator then adjusts the timeline according to the working team's consensus, and the ERs begin designing observations. During subsequent integration meetings, the observation timeline is adjusted based on design results and instrument team comments. Once the working team has developed an optimized timeline (approximately by the end of week 4), the science input portion of the Science Scoping process begins.

To make science sequence inputs, the ERs and SI use a software inputs package generically diagrammed in Figure 1 (Ref. 1,2). This software package creates a database which contains both scheduling and descriptive information for each science observation. The software is menu-driven and user friendly and can interact with mission-build software. The software uses the scheduling information (i.e., observation name, start/stop times, memory usage, scan platform usage, tape recorder usage, required telemetry data rate and format, rider instruments, etc.) to develop timelines and time-ordered listings, to perform resource summaries, and to identify timing conflicts. The software uses the descriptive information (i.e., the observation's science objectives, detail on how the observation is designed, details on how spacecraft memory usage is calculated, details on how tape recorder usage is calculated, etc.) to generate library and archival products. Once completed, the ERs use the software to deliver instrument observation information to a diskette. The technical support personnel then combine the ER's observation information with engineering, telemetry, and station coverage inputs from the SI and/or supporting sequence integration engineer to form a merged database. The software then uses this merged database to generate numerous specialized review products.

The working team generally has one review before the investigator review meeting. To perform this review, only a time-ordered listing, conflicts report, and certain resource summaries such as the tape recorder map, are needed. The working team resolves as many of the remaining conflicts as possible and documents them as resolved in the conflicts report. The ERs and science integrator then adjust their databases to reflect conflict resolutions and redeliver their diskettes to technical support. Technical support uses the updated database to generate the investigator review product. The investigator review product includes a time ordered listing, a time ordered listing sorted by instrument team, resource summaries which include a tape recorder map, a library describing the science objectives of each observation type, hardcopies of each observation's scheduling and descriptive information, and a timeline.

Following the investigator review, the working team incorporates investigator comments into the science scoping database. Technical Support then produces the final Science

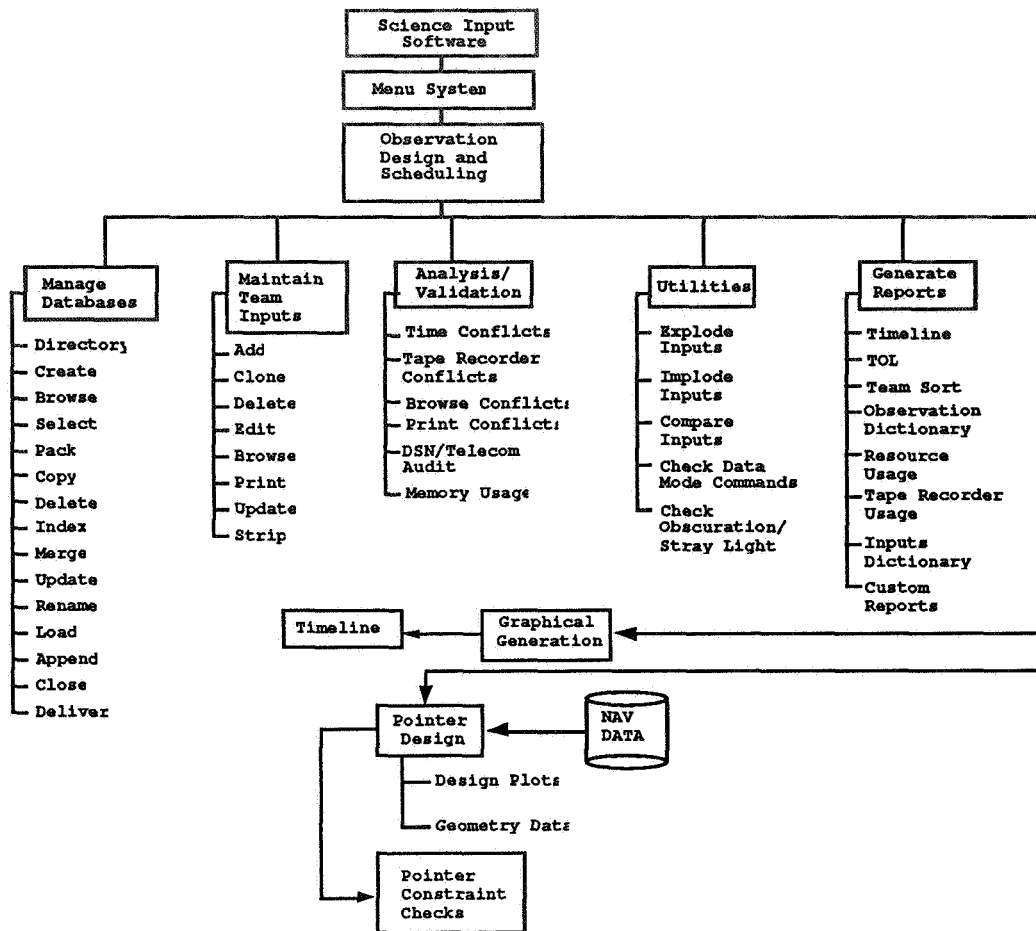


Figure 1. Functional Overview of Science Input Software

Scoping product while the ERs complete their observation designs. The final Science Scoping product is an updated investigator review product which includes a science overview description written by the science integrator. The Science Scoping product is delivered to and forms the baseline to begin the Sequence Plan step. By SP start, the ERs are also ready with completed or nearly completed observation designs.

3.2 Sequence Plan

The Sequence Plan step generally has two input or development stages. The process is approximately ten weeks long with the first input two to three weeks after SP start. At the first input stage, the ERs use the science input and mission design software to input their complete set of observation design parameters.

Once the ERs have delivered their observation design parameter inputs, the Sequence Integration Team combines science and engineering inputs into a complete timeline database. The sequence integration team runs the sequence through mission guidelines and constraints checking software and notes corrections which must be made by the science and engineering teams.

A key change to the Sequence Plan step is that, since the ERs have delivered completed observation designs, the SPOT can perform or work with the Sequence Integration Team to run detailed observation parameter check software and possibly spacecraft simulation software on all their science observations. If software complexity or the absence of initial condition and detailed engineering files do not allow spacecraft simulation software runs to occur at SP, the SPOT can develop specialized

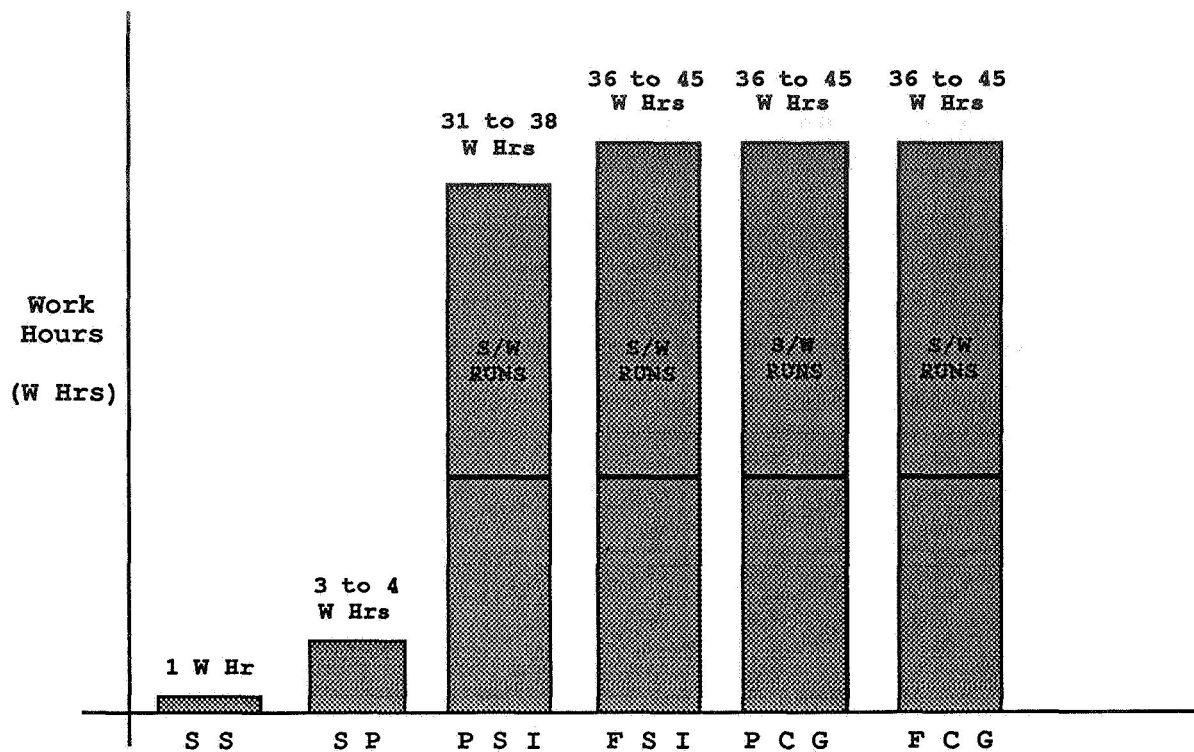


Figure 2. Estimated Work Hours for a Remote Sensing Observation Redesign/Reintegration

software which checks certain key science observation design parameters (i.e., scan platform slew start and end times, scan platform slewing rates when scan platform target motion compensation is active, and any other parameters which could require sequence reintegration if problems were discovered during spacecraft simulation software runs) at the same level of detail as the spacecraft simulation software. Hence, at this point in the SP process, before the sequence is under PCC, it is possible for the SPOT to discover and fix all possible science sequence problems. The fixes would simply be included in the SPOT delivery at the second or final SP input stage (approximately seven weeks into the SP process)

3.3 Cost Comparisons for changes at Science Scoping, Sequence Integration, and Command Generation Phases

Figure 2 displays cost comparisons for fixing a fairly standard integration problem at the different phases of sequence development. The integration problem selected for this comparison involves a remote sensing observation

slew overlap which requires either the observation to be shortened and/or redesigned, or for one or several surrounding observations to be moved. In all of these comparisons, assume that an initial product has been produced (i.e., an initial Science Scoping timeline, or an initial Sequence Plan or Preliminary Sequence Integration timeline) and that we are incorporating our changes into that product.

During the Science Scoping phase, the fix involves shortening or moving observation time locations and can be accomplished by merely changing the start/end times in the science input software. This change can easily be performed in thirty minutes. By allowing another thirty minutes to check the change in the mission design software, the total change can be accomplished in approximately one work hour.

During the Sequence Plan phase, the change is more involved since observation design parameters have already been input into the sequence. The individual(s) involved in fixing the problem would probably have to use the mission design software to redesign the

affected observation and to determine if any of the surrounding observations were adversely affected. They would then have to use science input and/or mission build software to change sequenced observation design parameters according to the fixes achieved in the mission design software. This observation redesign effort combined with observation parameter adjustments causes the cost of the fix to rise to three or four work hours.

At the Preliminary Sequence Integration (PSI) stage, all changes of this nature fall under Project Change Control, or PCC. Hence, the change request must be written on standardized mission request paper, by the sequence integration team, reviewed by the appropriate spacecraft engineering team subsystems to insure that spacecraft resources and subsystem health are not adversely affected, and submitted for review to a Project Change Board which includes sequencing and engineering team personnel plus the Mission Director and Office Managers. The observation redesign, paperwork, and requester and team chief checks cost an estimated three to four work hours. The sequence integration team impacts, spacecraft engineering team analyses, and team chief signatures cost an estimated six to eight work hours (Ref 3). The Project Change Board costs an estimated two work hours (twenty minutes for the presenter and each Change Board reviewer). A rerun of the mission's science observation and engineering event constraints checking software (based on the Galileo software model) could cost 20 to 24 work hours. Hence, the net total cost for this change at the Preliminary Sequence Integration could be as high as 31 to 38 work hours (Ref 3,4). The costs at Final Sequence Integration (FSI) and at the Preliminary and final Command Generation (PCG and FCG) levels would be the same as for the Preliminary Sequence Integration, except that now the mission's spacecraft simulation software may also have to be rerun. This rerun would add another five to seven hours to the total cost for a net result of 36 to 45 work hours (Ref 4).

4.0 SUMMARY AND CONCLUDING REMARKS

This paper has shown that there is a tremendous advantage for science to complete its observation design and integration (i.e., to effectively complete its work on the sequence) before the sequence goes under Project Change Control. The number of Project work hours involved in fixing an observation design

problem at Science Scoping versus after the sequence is under PCC can be a factor of thirty. By streamlining Science Scoping, limiting working team size, and using effective software tools, the SPOT can complete its observation designs and integration before the sequence undergoes PCC. By reducing, and by hopefully eliminating, the amount of time that science team members (and other Project personnel) are involved in fixing problems after the sequence has undergone PCC, more of the SPOT resources can be focused on exploiting science opportunities in other mission phases.

Since the early completion of the design and integration of science observations yields such potentially high cost savings, this paper would recommend that future Projects, when developing operating and mission software architecture plans, consider making the software sufficiently generic so that the complicated spacecraft constraint checking and modeling software can be run by the SPOT earlier in the sequence development process. If this approach proves infeasible, then it is recommended that future Projects consider developing specialized software packages that the SPOT could run to perform detailed constraint checks on selected observation design parameters at the Sequence Plan level.

The work described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

REFERENCES

1. Finnerty, D.F., Martin, J., Doms, P.E., 1987. "Asset: An Application in Mission Automation for Science Planning", JBIS, 40, 461-470.
2. Henderson, V., private communications.
3. Tan, K., private communications.
4. Thomas, R., private communications.

A PACKET-BASED CONCEPT FOR SPACECRAFT COMMAND PLANNING

N 9 4 - 2 3 8 7 3

Valerie B. Barnes

The Johns Hopkins University Applied Physics Laboratory
Laurel, Maryland/USA

ABSTRACT

The current generation of spacecraft being developed and operated by the Applied Physics Laboratory provides users with access to a broad spectrum of scientific instruments on maneuverable platforms that can be oriented for observation of both moving and stationary targets of interest. The capability of these increasingly complex spacecraft to perform data collection operations is approaching one observation per orbit. To enable both rapid configuration and generation of complex spacecraft command sequences, as well as reusability of command sequences among data collection operations, a packet-based concept for spacecraft command planning has been developed.

The configuration of the spacecraft for any operation is designed using "packets" where a packet represents a set of commands that is reusable. The packets can be combined in varying levels of functionality, and in varying time relationships, to create an observation timeline. At the lowest packet level are primitives. Primitives relate the details of command generation for a particular spacecraft to a "message template." Thus the packet concept itself is reusable from one spacecraft to the next.

Key Words: Operations planning, command, aerospace

1. INTRODUCTION

Complex satellites are built to support many different experiments. Because the same instruments and spacecraft subsystems are being used for those experiments, there is some degree of command commonality across different experiments. Often, a single

experiment cannot be completed by one data collection event; rather, the experiment is designed as a series of several similar data collection events. The operations planning process developed at APL is designed to exploit command commonality across experiments and within each experiment class.

Spacecraft events are carried out by executing a series of commands that tell the spacecraft how to maneuver, what devices to use, how to configure them, and when to take each action. To streamline the event planning process, sets of spacecraft configuration information are defined. This configuration information is divided into two types: reusable building blocks called **planning packets** and event-peculiar structures called **event specifications**.

Figure 1 shows possible planning packets. The planning packets are used to simplify the planner's job. For instance, a specialist may make a low-level packet to partially or completely define a particular instrument configuration for several experiments. A mid-level packet may be set up to accomplish one type of maneuver, leaving details such as pointing angles and timing blank. Or, a packet may be constructed to define a spacecraft-wide function such as recording data in one of several formats.

A planner builds a top-level packet from existing lower-level planning packets to meet the sequence and configuration requirements for an experiment type rather than starting from scratch each time. When a data collection event for that experiment is scheduled, the planner sets up an event specification which points to the top-level packet.

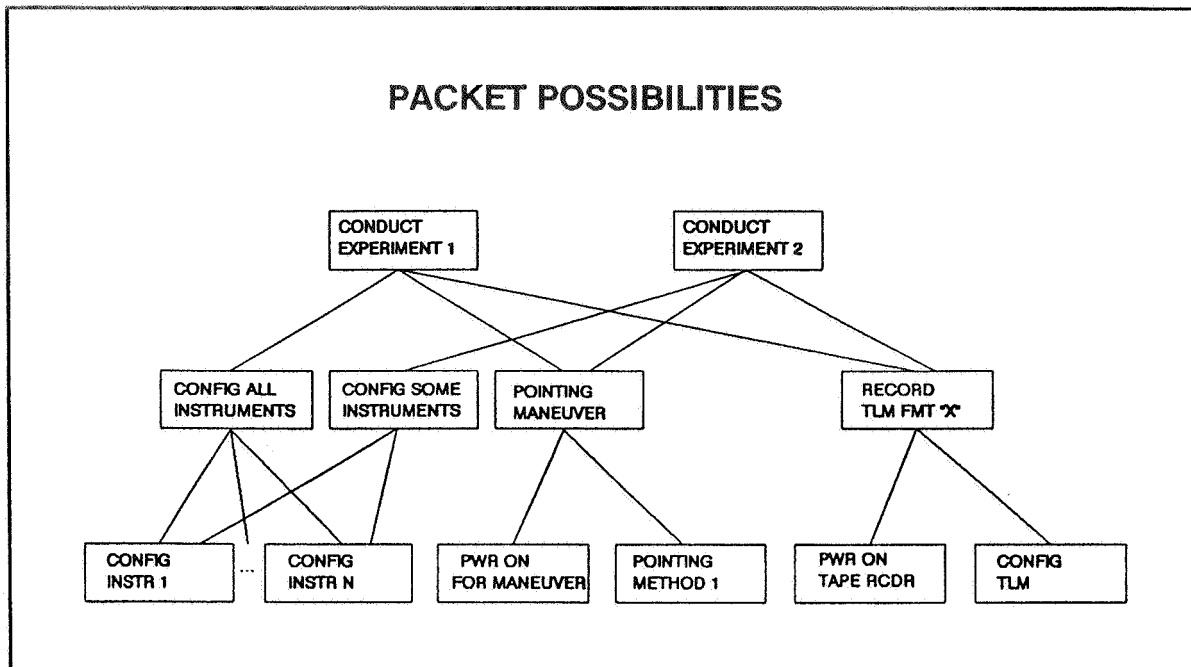


Figure 1

All the data and timing from the pre-built packet are used to initialize the event specification. The packet data are tailored for that particular data collection event by filling in each blank data value, adjusting timing, and storing the results in the event specification. When completed, the event specification contains all of the data necessary to formulate all the spacecraft commands required for the event.

2. RAW MATERIALS FOR PACKETS

APL satellites are operated via "messages," each consisting of one or more commands. In the operations planning system, **message templates** are used to define the characteristics of each message type. The information stored for each message template is used in building the reusable packets; in soliciting, validating, and interpreting event specification data; and in formulating messages in the spacecraft's command language (APLCOM). The message template library contains the following for each message template:

- Message Template ID
- Message Template description

- Command type (serial digital, pulse, relay, memory load)
- Number of data fields
- APLCOM format
- For each data field
 - name
 - description
 - type & length
 - category (list, constant, range, calculated)
 - units
 - default value (optional)
 - list of acceptable values or value range and accuracy
 - power delta corresponding to each "mode" change
 - locked command flag
 - telemetry response.

Thus, message templates form the foundation for the packet-based operations planning system.

Each message template only designates the valid alternatives for its data fields. To give a message template meaning for a particular function, at least one data field must be filled in. The next layer in the operations planning hierarchy is a **primitive**.

A primitive is a reusable instance of a message template in which at least one data field is filled in. A primitive can be defined to configure a specific device (e.g. channel 33 relay ON) or to configure a generic device (e.g. tape recorder "X", record at 25 Mbps). The user identifies the data fields in the primitive as constant or variable and sets fixed or default values in data fields. Making every data field in a primitive a constant means that no additional information will be entered during the event specification process.

Primitives are stored in a library for use in planning packets.

3. PLANNING PACKETS

There are three types of planning packets in the packet hierarchy: **event packets**, **action packets**, and **command packets**. Figure 2 depicts the packet hierarchy.

The lowest-level packets are the **command packets**. Command packets define reusable sequences of primitives in which timing is fixed for a particular component, sensor, or group of spacecraft components. These packets are used to perform some basic function such as a power-on sequence.

At the next level, **action packets** are used to define configurations that accomplish some broader function common to more than one event. Action packets are constructed from command packets. An action packet might be used to define a type of maneuver or a series of filter setting changes for data collection with an imager.

Event packets occupy the highest level of the packet hierarchy. Event packets are collections of action packets which define configuration sequences for every spacecraft component needed to perform an event.

Packets contain this information:

- o Packet ID
- o Packet name
- o Packet description
- o Spacecraft subsystem, instrument,

subsystem component, instrument component or sensor

- o Number of packet elements
- o For each element of the packet
 - element ID (primitive or lower level packet)
 - element sequence number.

When a packet is defined, the user may set defaults (data values and time offsets). The defaults are for:

- o primitive data field values
- o device selections
- o definition of a scheduling point for the packet
- o relative time offsets from that scheduling point for everything else in the packet.

Some defaults may be defined to be constants; others may vary. A variable data field value may be changed at any higher packet level or in the event specification. Invariability flows up the packet hierarchy (primitive to command packet to action packet to event packet). The packet defaults are used as the starting points for an event specification.

Every packet must have exactly one scheduling point (time offset 0). The scheduling point is used to schedule a command packet in an action packet or an action packet in an event packet. If necessary, the user may use a dummy placeholder for the scheduling point. The placeholder will not be translated into a spacecraft command; it will only be used to establish relative timing within the packet.

Timing relative to the scheduling point is set for each element in the packet. If the element is itself a packet, the time offset applies to all internal sub-elements. Relative time offsets must be defined as constants in every command packet. Relative timing defaults may be assigned for higher-level packets; if they are set, they are variable. Packets are stored in libraries for reuse.

4. EVENT SPECIFICATIONS

The **event specification** is created when an

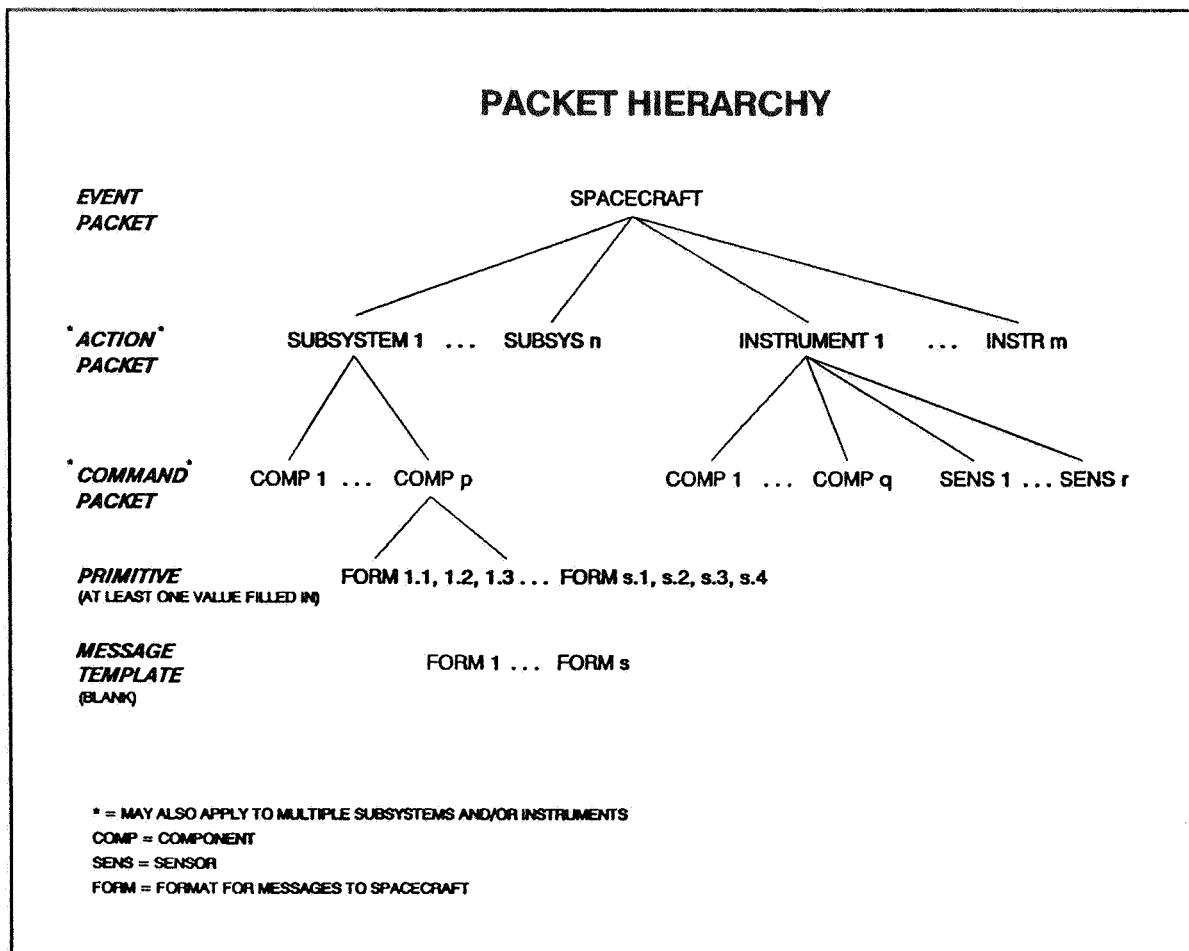


Figure 2

event packet is linked to a particular data collection, maintenance, or a downlink event. The constant, default, and timing values associated with the planning building blocks comprising the event packet are then copied into the event specification. An event packet must exist before the event specification can be created. Figure 3 illustrates the relationships among packets and shows the kinds of data associated with an event specification. The event specification data may be divided into these categories:

- Event-level data
 - planned T-0 (time used in analysis of the event)
 - scheduled T-0 (time at which the event has actually been scheduled)
 - description
 - relationships to other events

- Relative time offsets
- Primitive data values.

The event-level data in the event specification are unique to a particular event. The remaining data are initialized based on the values set in the default section of the packet. The defaults were first set as constants or variables during the packet-building process. The variable settings may be edited. The user is prohibited from changing any packet data that were set as constants. Relative time offset data tied to packets and the event's T-0 are used by analysis and command building software to determine the sequence and absolute execution time of primitives over the entire event.

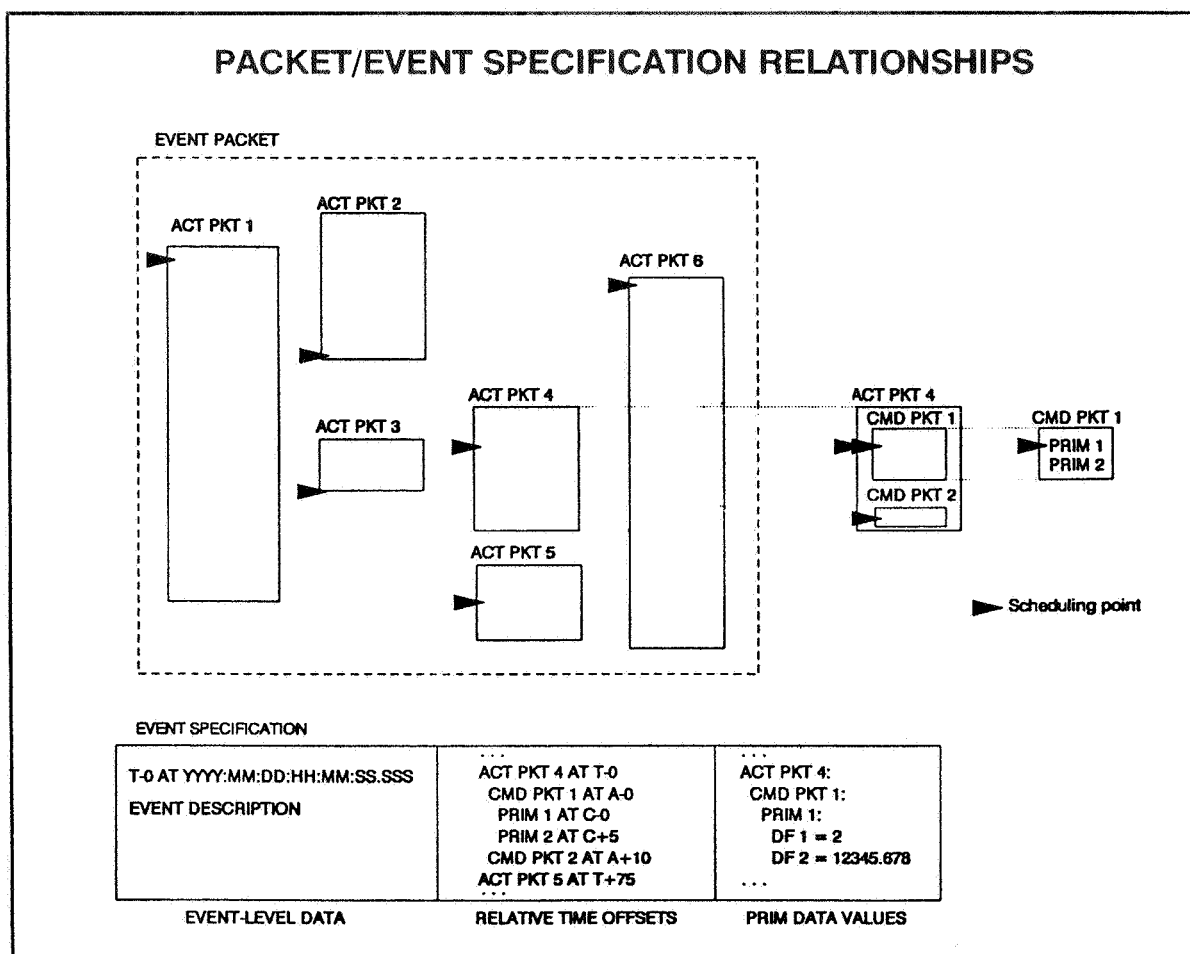


Figure 3

5. IMPLEMENTATION

The event specification is used to drive event analysis software. This software evaluates the predicted geometry, attitude, power loads, and thermal conditions. Event analysis software also uses the event specification to assess compliance of the event's sequence of activities with flight rules, and verifies that all elements of the specification are consistent.

The command formulation software builds commands based on the data in the event specification and on the APLCOM format information stored for each message template.

All packet and event specification data are stored in a relational database. Point and click windows function as user interfaces for

building primitives and packets, editing event specifications, executing analysis software, and formulating commands. Software is designed to check a user's input against data value domain definitions stored in the database. The system is implemented on a network of RISC workstations operating under UNIX.

6. CONCEPT PROS/CONS

6.1 Pros

The packet concept enables operations planners to catalog knowledge about how to use the spacecraft and to reuse that information for a similar purpose. Packets can be built in advance and checked once. Packets can be set up to mirror the physical structure of the spacecraft or to match the logical nature of a function spread across

several subsystems.

Software is used to verify the activities in an event sequence prior to building commands. Software is also used to build commands. By removing humans from the command formulation step, translation and transcription errors are eliminated.

The packet-based planning concept and system design are transferrable to other spacecraft. For a different satellite, a new database of message templates and libraries for primitives and packets would be needed.

6.2 Cons

Using packets to plan is an effective solution if the spacecraft performs similar functions repeatedly. The packet-based system is cumbersome when building small command sets for one-time use.

To build packets that are reusable, the builder must have a broad understanding about how different experimenters plan to use the spacecraft. APL addresses this issue by assigning responsibility for the smallest building blocks to spacecraft specialists who are very familiar with the spacecraft. Operations coordinators who understand all experiment plans within one or more experiment classes are responsible for higher-level planning packets. Planning team members communicate often so that multiple packets are not built for the same purpose. One team member is responsible for maintaining control of the packet libraries.

6.3 Summary

The packet-based concept for command planning provides flexibility and reusability to the planning process. It allows the analysis and command-building processes to operate from a common set of information. Using this concept, planners think in terms of how to command the spacecraft in every planning phase. The on-orbit planning team size can be minimized through the use of pre-built planning packets.

ACKNOWLEDGMENT: This concept was developed to support the operations planning function for the Midcourse Space Experiment under Navy contract N00039-91-0001. Turning the packet concept into reality would have been impossible without the brainpower and dedication of my colleagues. Special thanks go to our database guru, Ron Glaser, whose clarity of thought, insistence on consistency, and sense of humor were vital.

CONNECTING SCIENCE AND OPERATIONS: THE OPERATIONS COORDINATOR

N94-23874

Madeleine H. Marshall and John A. Landshof

The Johns Hopkins University Applied Physics Laboratory
Laurel, Maryland/USA

ABSTRACT

For a current space mission under development at the Applied Physics Laboratory, the Mission Operations staff includes a team of "Operations Coordinators" who have been working with the Mission Science and Spacecraft Development teams since completion of mission conceptual design. The Operations Coordinators are responsible for bringing knowledge of the spacecraft to the Mission Science team, and bringing knowledge of the experiment requirements to the Spacecraft Development and Mission Operations teams. Once on-orbit operations begin, the Operations Coordinators will be responsible for implementation of specific science experiments from analysis through scheduling, generation of spacecraft command sequences, delivery of science data to the end user and operations assessment. The Operations Coordinator concept is proving very effective during the development phase of the current mission.

1. INTRODUCTION

From March through December 1989, the "Delta Star" spacecraft provided scientists with a multi-spectrum suite of scientific instruments in low earth orbit that could be positioned for observation of both moving and stationary targets of interest. Science data recorded on flight recorders were downlinked to ground stations, made available for review and analysis at a science data center, and then forwarded to a central archive for future use. During the on-orbit phase of operations, over thirty Principal Investigators proposed experiments for the Delta Star spacecraft. Many of these experiments, which evolved as the mission progressed, involved cooperative ground operations. For the Mission Operations team to effectively implement these experiments and make efficient use of spacecraft and ground resources, it became clear that each Principal Investigator needed a partner in Operations who would be responsible for their experiment from inception through delivery of data to the end user. This "partner" needed both a clear

understanding of the spacecraft and ground system capabilities as well as the technical background necessary to grasp the science objectives and translate them into spacecraft and ground system operations. Additionally, the "partner" would provide a single point-of-contact in Mission Operations for a Principal Investigator and, as a member of the Mission Operations team, could provide schedule coordination for experiments that required the use of other ground or airborne sensors in conjunction with spacecraft observation of a particular phenomena. Thus, the concept of an Operations Coordinator emerged.

Informal tests of the Operations Coordinator concept during the latter stages of the Delta Star mission indicated that, not only was the concept effective, but it also provided very interesting and challenging work for senior level operations staff members. The decision was made to implement this concept at the outset for the next mission that resembled Delta Star in terms of spacecraft complexity and user variety. The Mission Operations system for the current space mission under development employs the Operations Coordinator concept.

2. THE MISSION OPERATIONS SYSTEM

The Mission Operations system consists of the ground system, the Mission Operations teams and the processes in which the teams and the ground system engage to operate the spacecraft. The "concept of operations" describes the Mission Operations system teams and processes based on the ground system design. Functionally, the system is divided into three major areas: Operations Planning, Operations Control and Operations Assessment. The Operations Planning function is concerned with analyzing and scheduling spacecraft operations and generating command sequences; the Operations Control function is concerned with communication of command and telemetry data to and from the spacecraft and data distribution; and the Operations Assessment function is concerned with evaluation of the performance of the spacecraft

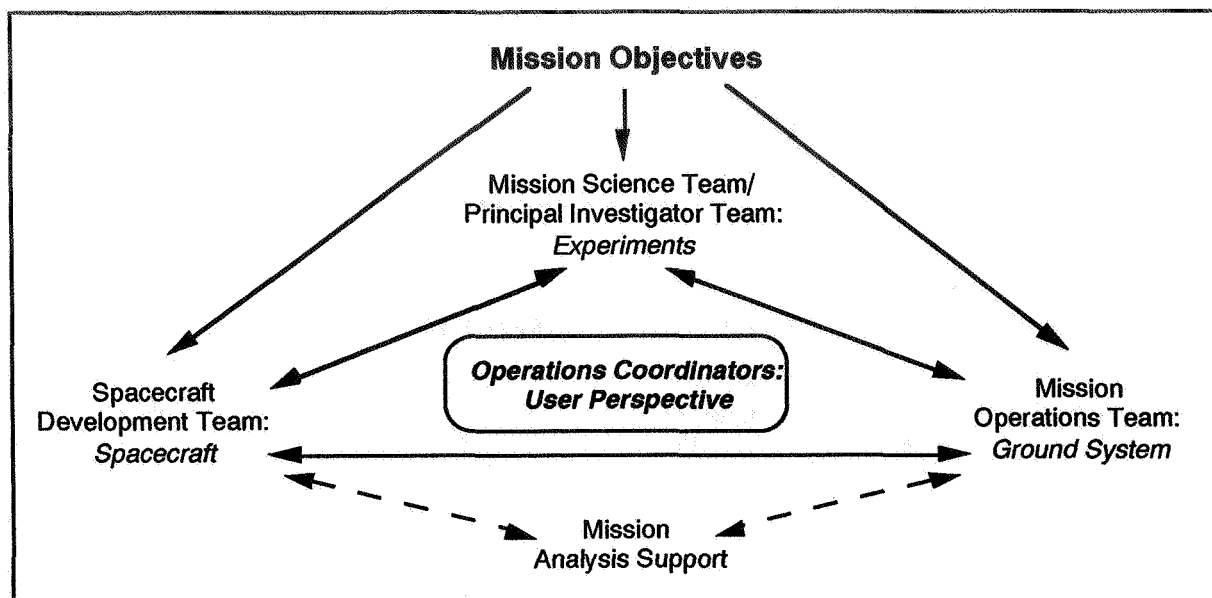


Figure 1. Operations Coordinator Role in Mission Design

and of the Mission Operations system. Each functional area is the responsibility of an operations team; each team is supported by elements of the ground system.

The Operations Coordinators are considered members of the Operations Planning Team. For the current mission, this team includes four Operations Coordinators representing eight Principal Investigator teams, and eight "Operations Analysts" who provide analytical support for operations planning activities. Each Operations Coordinator is assigned from one to three Principal Investigator teams depending on commonality of science objectives.

3. THE OPERATIONS COORDINATOR ROLE IN MISSION DESIGN

The space mission design process (Figure 1) requires the balancing of mission objectives with spacecraft system capability and ground system capability. For a scientific mission, translation of mission objectives into experiments or tests that will use the spacecraft and ground system is the responsibility of the Mission Science, or Principal Investigator team. The Spacecraft Development team and the Mission Operations team develop complementary system designs that will support execution of the experiments or tests proposed by the Principal Investigator team. Both spacecraft and mission operations engineers require analytical support to interpret science requirements in terms of system capability and to set design parameters.

Although the role of the Operations Coordinator as it evolved during the Delta Star mission concentrated on the on-orbit interface between science and operations, this connection is essential from the beginning of the mission design process. Typically the Mission Operations team that is identified at this time includes a ground system engineer with some level of mission analyst support. All team members are charged with understanding the science objectives as they apply to their system. However, properly defined, the Operations Coordinator role can bring a unique perspective to this process—that of the real user of the spacecraft and the ground system. If the Operations Coordinators are charged with the responsibility of translating science experiments into operations and following those operations through delivery of data back to the science teams, then they must be concerned with the integrated usability of the spacecraft and ground systems in the execution of science experiments.

For the current mission, eight Principal Investigator teams were chartered early in the mission design process, each with their own class of targets or phenomena to be observed. Two Operations Coordinators were designated as part of the core Mission Operations team and charged with taking the system "user" perspective as they developed both an understanding of the Principal Investigators' experiment proposals and the design of the spacecraft and ground systems. In this role they provided the user point-of-view in design

sessions and were also tasked with developing the "concept of operations" for the current mission.

4. THE OPERATIONS COORDINATOR ROLE IN SYSTEM DEVELOPMENT

Following the conceptual design reviews for both the spacecraft and ground systems, core members of each operations team were named to assist the ground system engineer in developing detailed specifications for the ground system elements that would be used by that team. As spacecraft system development began in earnest and the Principal Investigator teams actively moved forward with experiment scenario design, additional Operations Coordinators were identified to handle an expanding role. For each Principal Investigator team, the Operations Coordinator was tasked with instructing the team in the use of the spacecraft and, for each proposed experiment, assessing whether or not that experiment was feasible. This process (Figure 2) is iterative, as modifications to the experiment, the spacecraft, and the ground system are often possible in the early development stages.

An experiment is considered to be feasible if a spacecraft operation can be created and scheduled that responds to the objectives of the experiment and where spacecraft resource usage for that operation is within allowable limits. The tools needed by the Operations Coordinators to evaluate experiment feasibility are the same tools as are needed by the Operations Planning team on-orbit to configure and schedule the spacecraft and manage spacecraft and ground resources. These tools include orbit propagators, spacecraft dynamics models, tools to analyze satellite-to-target geometry, spacecraft configuration tools, and spacecraft resource usage models. Therefore, a second role of the Operations Coordinator during system development is to support the ground systems engineer in specifying user requirements for system capability. As early users of the ground system elements that support analysis, the Operations Coordinators also serve as members of the system test team.

5. THE OPERATIONS COORDINATOR ROLE ON-ORBIT

For the current mission, it is anticipated that the majority of experiment design will be completed prior to launch. Although some level of experiment design and feasibility analysis will

continue after launch, the primary role of the Operations Coordinators will shift at launch from experiment and ground system design support to on-orbit operations coordination. At launch, the full Mission Operations team is in place. To understand the Operations Coordinators role on-orbit, it is necessary first to understand the baseline process of scheduling and executing satellite operations for the current mission (Figure 3).

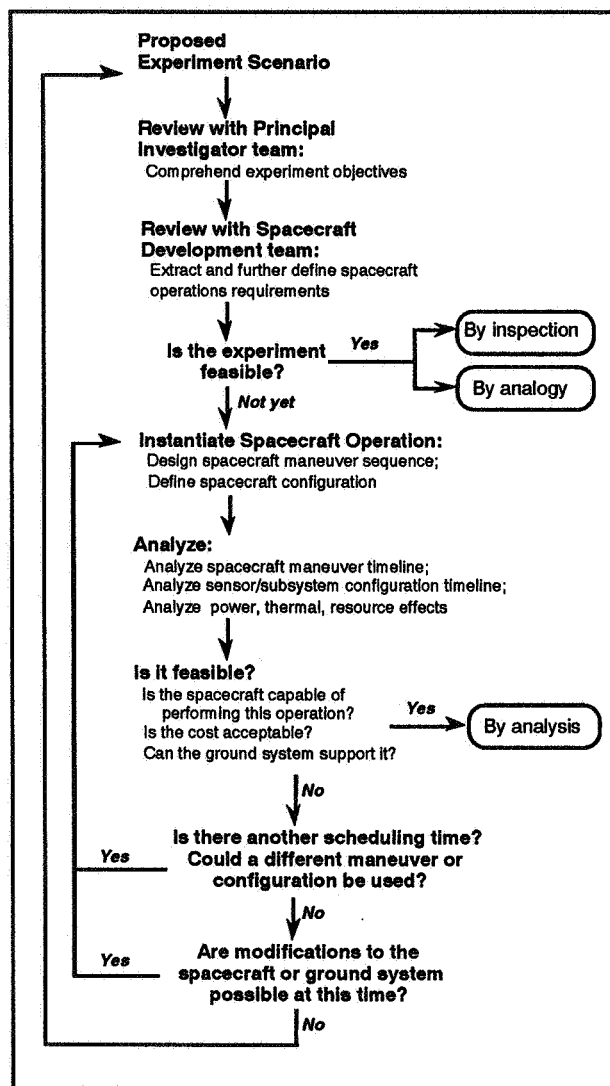


Figure 2. Experiment Feasibility Assessment

On a monthly basis, a set of experiments are selected by the Mission Director (representing the sponsoring agency) and provided to the Mission Operations team as "monthly objectives". This occurs approximately six weeks prior to the beginning of the month of interest. The Operations Planning team identifies the scheduling opportunities for the

spacecraft operations associated with each experiment and interleaves the operations into a monthly schedule accounting for experiment priority, spacecraft recovery between operations and data downlink requirements. The ground station contact activities are coordinated on a weekly basis and reported in a weekly schedule. Each day the schedule for the following day is refined; objectives are defined for each contact the ground network makes with the spacecraft, a daily schedule is assembled detailing all contact activity and spacecraft activity, and commands are generated. Contact level procedures, called contact plans, are generated specifying what action is to be taken and what critical spacecraft status is expected on each contact, and then delivered to the control teams for execution.

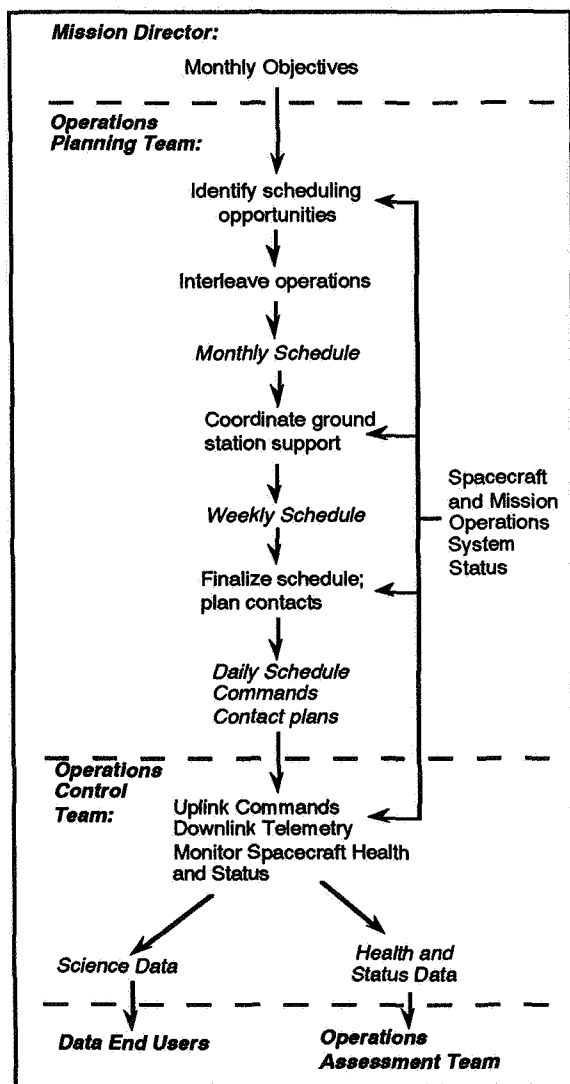


Figure 3. Baseline Operations Process

The Operations Control team responds to the contact plans provided by the operations planners. The contact plans provide details on the link configuration, tracking and downlink activity, uplink requirements and spacecraft status. For each contact, the controllers configure the data links, establish communication with the spacecraft, uplink commands as required and manage telemetry downlink and tracking data acquisition, monitor routine spacecraft health and status, and document the actual contact. For those contacts where science data is downlinked, the Operations Control team is responsible for data dissemination to the end users.

The Operations Assessment team analyzes and reports on the execution of each spacecraft operation based on the planned operations specifications and downlinked telemetry data.

The Operations Coordinators are responsible for implementation of specific science experiments from analysis through scheduling, generation of spacecraft command sequences, delivery of science data to the end user and operations assessment. In particular, they direct all operations analysis done at the monthly and weekly scheduling stages. They are supported in this work by the Operations Analysts. Each Operations Coordinator represents the interests of his/her Principal Investigator teams during scheduling exercises. The Operations Coordinator can make changes to experiment operations within guidelines provided by the Principal Investigators if this can improve on the quality or volume of data collected. If an experiment requires coordination with other ground or airborne sensors, the Operations Coordinator provides scheduling information as needed. For each operation, the Operations Coordinator monitors all daily scheduling, command generation and operations control activities performed in support of that operation, provides status to the Principal Investigator as needed, and signs off on the report made by the Operations Assessment team about that operation. Finally, the Operations Coordinator provides the single point-of-contact in Mission Operations for the Principal Investigator concerning his/her experiment operations on-orbit.

6. STAFFING CONCERNS

The broad nature of the Operations Coordinator job requires a broad skill base. The Operations Coordinator must develop an in-depth

knowledge of both the spacecraft and the Mission Operations system, as well as a clear understanding of the objectives of the experiment plans. Although analysis support may be provided by the Operations Analysts, the Operations Coordinator must be able to direct the analysis and interpret results in order to assess whether or not an experiment is feasible, what the resource impact on the spacecraft is, and to select which scheduling opportunities for an experiment may be most productive. Excellent communication skills are essential. Operations Coordinators brought in as part of the core Mission Operations Team during mission and system design have the advantage of learning the systems as they are developed; for all Operations Coordinators the ability to take a system perspective is mandatory.

7. IN PRACTICE

Design of the Mission Operations system for the current mission was initiated three and a half years ago and the system is expected to be operational late next year. Early identification of Operations Coordinators has provided significant gains in the quality of the Mission Operations system being developed:

- the Operations Coordinators have provided continuous input into the experiment design process concerning the capability and limitations of the spacecraft and the ground system;
- the Operations Coordinators have provided continuous input into the spacecraft development process concerning the experiment requirements and the proposed spacecraft operational scenarios;
- the concept of operations addresses the fundamental purpose of the Mission Operations system to collect science data;
- the ground system element that is being developed to support operations analysis is tailored to the types of experiments that will require analysis;
- an effective, productive working relationship exists between the Mission Science and Mission Operations teams; and
- Mission Operations is viewed as being a responsive, responsible organization.

THE "INSTANT SEQUENCING" TASK: TOWARD CONSTRAINT-CHECKING A COMPLEX SPACECRAFT COMMAND SEQUENCE INTERACTIVELY

Joan C. Horvath, Leon J. Alkalaj, Karl M. Schneider, and Arthur V. Amador

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109
USA

N94-23875

Joseph N. Spitale
California Institute of Technology
Pasadena, CA 91125
USA

ABSTRACT

Robotic spacecraft are controlled by sets of commands called "sequences." These sequences must be checked against mission constraints. Making our existing constraint checking program faster would enable new capabilities in our uplink process. Therefore, we are rewriting this program to run on a parallel computer. To do so, we had to determine how to run constraint-checking algorithms in parallel and create a new method of specifying spacecraft models and constraints. This new specification gives us a means of representing flight systems and their predicted response to commands which could be used in a variety of applications throughout the command process, particularly during anomaly or high-activity operations. This commonality could reduce operations cost and risk for future complex missions. Lessons learned in applying some parts of this system for the TOPEX/Poseidon mission will be described.

Key Words: Sequencing, mission operations, automation, parallel computing.

1. INTRODUCTION

Robotic spacecraft are controlled by sets of time-tagged onboard commands called "sequences." These sequences must be verified and checked against mission constraints to be certain that they will have the planned effect on the spacecraft. One of the key automated portions of the sequence verification process is a checking program which does some limited functional simulation of the sequence running on the spacecraft and compares the resulting spacecraft states against mission rules. Constraint-checking occurs at many points in the so-called "uplink process", which creates spacecraft command sequences. Checking occurs at the command timing

interaction level, but also at the high-level planning and sequence-integration stages.

Making our existing sequence-checking program, (called the "Checker") faster would make some changes in our uplink process, since then people building command sequences would not tend to check a sequence by hand before submitting it to lengthy batch runs. We determined that a good way to make the Checker code run faster would be to run it on a parallel computer, which meant that we had to determine how to run constraint-checking algorithms in parallel. Directly porting the existing Checker code to a parallel machine, however, proved to be difficult due to the inherently non-parallel way in which the flight rules and the spacecraft models on which they acted were encoded [Refs 1-2.] Once we realized this, we concentrated our efforts on exploring better ways of specifying constraints and models.

When we created this new specification system, we realized that we had a way of describing flight systems that was more broadly applicable than just within the traditional "sequence checking" part of the flight command process. A general representation of flight systems and their predicted response to commands could be used in a variety of applications throughout the uplink process, particularly during anomaly or high-activity operations. We currently have two development efforts under way: a system intended to run on a parallel computer, and an operational (sequential) system for use by the TOPEX/Poseidon spacecraft.

2. SPECIFICATION AND VERIFICATION

2.1. Basics

We describe spacecraft systems in terms of three

fundamental pieces: the rules to be checked, a model of the subsystem(s) of the spacecraft or ground system to which the rule(s) apply, and an “action table” that describes interactions among the models and illegal transitions inside the models [Ref. 3]. It is also very desirable to have “status events” that flag, in user-readable form, changes of state of some spacecraft components. Each of these is now described in turn.

2.2 Rules

When a spacecraft is designed, “flight rules” or “mission rules” are developed to prevent damage to the hardware, to prevent loss of science data, or to simplify and constrain the operation of the spacecraft.

In our system, which we call SAVE (Specification and Verification Environment) each rule to be checked by software is expressed as a logical constraint over state transitions. The constraint can be of temporal nature as well as a constraint over state orderings. Informally, the syntax of a rule is as follows.

*Whenever (a state -> a certain value)
if (some condition holds)
=>(a violation of the flight rule has occurred).*

Where the “->” symbol should be read as “goes to” and the “=>” value should be read as “generates.” The states and models used in the rules are defined in *models* and *action tables*.

2.3 States

The states may be in the format

model.state

where *model* is the name of the model to which the state belongs, and *state* is the particular state variable. For temporal comparisons, the state may also be shown as:

model.state.time

where the *time* field is the time at which a given state achieved its most recent value. The syntax used inside the “if” clauses of the rules is the standard syntax for a logical expression in the “C” programming language; e.g. “&&” for “AND”, “||” for “OR”, and so on.

2.4. Status information

It is desired that the status of certain variables be printed out whenever the state changes (even if the change is not a flight rule violation.) The syntax for status events is similar to that for rules, namely:

*Whenever (state-a goes to a value)
if (condition)
then status (state-a,state-b,state-c....)*

where *state-a* is the “trigger state” which will cause a status message to be generated, and *state-b, state-c, ...* are states the user may want to see as well when *state-a* changes. Usually the condition in the “if” statement will be TRUE unconditionally -- that is, the status event will always be printed out irrespective of any other states.

2.5 Models

Every rule and status event requires that one or more “models” of a limited subset of the spacecraft behavior be generated. These models can be shown in “finite-state” form: that is, a portion of the spacecraft is modeled in terms of several discrete variables (an “A” and “B” redundant side, for example). The commands or other actions that cause the system to transition from one state to another are shown on the arcs between the commands. [Figure 1].

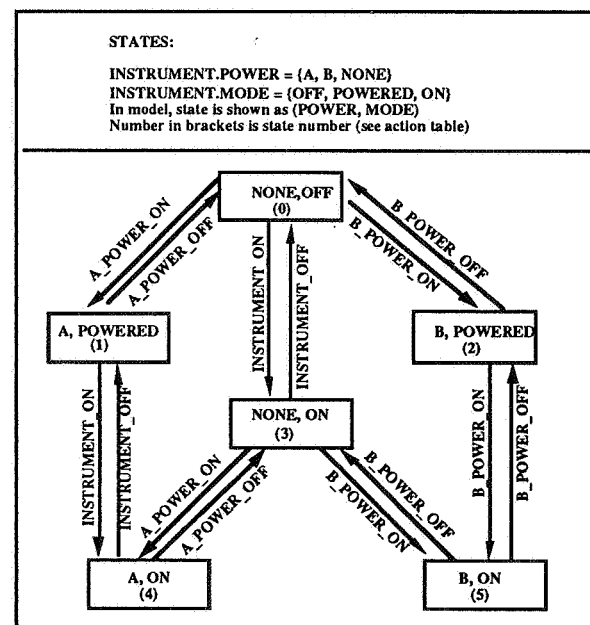


Figure 1. A typical state diagram.

Figure 1 models an instrument on board a spacecraft which can be powered by one of two redundant power supplies - A or B. The command to close the relays from the instrument to the A power supply is the "A_POWER_ON" command; similarly with the "B_POWER_ON" command. Once the instrument is powered, it can be turned on to its operational mode with the "INSTRUMENT_ON" command. This model shows that the instrument can be turned on when it is unpowered, but it will then surge when power is applied. This "illegal transition" is discussed in the following section.

2.6 Action tables

The finite-state models can also be shown more completely in a spreadsheet-like form. This form has all the legal states in the vertical direction, and all the transition commands in the horizontal direction. Every command is analyzed for its actions should it be issued in any of the states. In some cases this will work out to nothing happening; in others this command will cause an illegal transition. These illegal transitions will also be flagged by the checking software if and when they occur. The "next state" field in each box of the action table tells the software what the next state would be should the relevant command be issued.

Figure 2 shows an "action table" for the system shown in Figure 1. The blank boxes in the Action rows imply that the transition simply takes place to the "Next State" shown and no side effects occur. The "ERROR" states shown (for example, the transition from State 0 to State 3 caused by the INSTRUMENT_ON command issued when the spacecraft is in State 0) would generate an "illegal transition" message to the user. Arbitrarily complex "side effects" (e.g., effects on other models) can appear in these "action" boxes in the form of function calls in the "C" programming language.

NO.	POWER	MODE	COMMAND->	A POWER ON	B POWER ON	A POWER OFF	B POWER OFF	INSTRUMENT_ON	INSTRUMENT OFF
0	NONE	OFF	ACTION					ERROR	
			NEXT STATE	1	2	0	0	3	0
1	A	POWERED	ACTION		ERROR				
			NEXT STATE	1	1	0	1	4	1
2	B	POWERED	ACTION	ERROR					
			NEXT STATE	2	2	2	0	5	2
3	NONE	ON	ACTION	ERROR	ERROR				
			NEXT STATE	4	5	3	3	3	0
4	A	ON	ACTION		ERROR	ERROR	ERROR		
			NEXT STATE	4	4	3	3	4	1
5	B	ON	ACTION	ERROR		ERROR	ERROR		
			NEXT STATE	5	5	5	3	5	2

Figure 2. An Action Table.

Let's say that there is a restriction on the system shown in Figures 1 and 2 that will not allow other instruments to be turned on if this instrument happens to be turned on. A typical rule, then, for the system described in Figure 1 might then be expressed as:

*Whenever (INSTRUMENTMODE -> ON)
if (OTHER_INSTRUMENTMODE == ON)
=> violation.*

Note that there is no hard-and-fast distinction between an "illegal transition" and a "flight rule"; some illegal transitions have been called out as particularly troublesome and made rules. Implementationally, some are easier to call out one way and some in others.

3.0 PARALLEL IMPLEMENTATION

3.1. Overview

Our next step was to build a small prototype system that used these principles and implemented them on a parallel computer. This SAVE software implementation consists of two major functions. The primary function is sequence verification; the second (related) function is system specification. System specification is a kind of "installation" function which normally would not be used routinely. In system specification, an operator using finite state machine notation lays out a spacecraft/ground model, and specifies constraints upon the behavior of this model. System verification is "production" running of the system to check a sequence with pre-defined constraints.

The system was designed in three basic parts: the "core", the "compiler" and the "database." [Figure 3]. The compiler takes as input rules in the "whenever" syntax described above, and models in a database.

format derived from the action table format shown in Figure 2. When a user adds a model or rule, these inputs are compiled into in-line C code, which is linked with the core code.

3.2. Runtime environment

At runtime, the core code determines from a user input how many processors and models it will have for the given run. It assigns the models to processors according to a user-specified definition. A sequence of commands to be checked against rules is read and stripped of all information that is not relevant to the sequence rule checking function. The commands in the sequence are then partitioned out to the different processors according to the model to which they "belong."

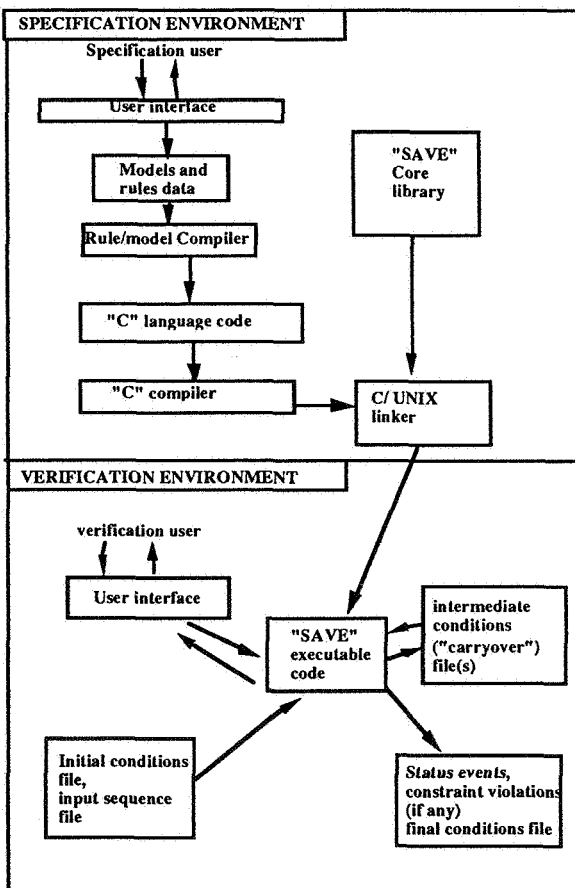


Figure 3. Architecture summary.

Then, starting from an initial state for each model, each command's effects on the system are simulated according to the data in the action tables. If a state changes that triggers a rule check, the rule is activated

and an error message, if any, is output to the error log. A final state is written to a file at the end of the run.

3.3. Parallel processing results

A prototype was built on a five-processor parallel computer. In this prototype, a "conservative" synchronization approach [Ref.4] is used to ensure that processors do not get out of synchronization with each other. This prototype showed that this methodology could achieve excellent speedup for this type of problem depending heavily upon the actual models and sequences being checked. (Models which interact with each other frequently make less efficient use of the parallel computer. Sequences which use one model disproportionately make less efficient use, as do very short sequences where work tends not to average out over the parallel processor too well.)

The methods used in the parallel prototype could easily be adapted to a distributed system or to any of a variety of parallel architectures. The system is best suited for the type of parallel processor which has substantial memory and processing power on each node (i.e., a "large grained" or "medium grained" machine). We also use this parallel system as the basis for our one-processor system we are implementing for the TOPEX/Poseidon project. For the one-processor version, we have built routines that mimic the parallel interprocessor communications and return the same value(s) as would the parallel one. Since most of these routines do nothing on one processor, this was not a complex task.

4.0 LESSONS LEARNED FROM FLIGHT IMPLEMENTATION

4.1. Moving a prototype into a flight environment

Recently it was decided to build a sequence-checking system for the TOPEX/Poseidon spacecraft based on the SAVE prototype system. This full system is called "MSAVE" (for Mission Planning, Sequencing and Scheduling Specification and Verification Environment.) Several modifications were necessary to the system for it to be used in the TOPEX environment [Ref.5].

4.2. Modifications required

These modifications fell into three basic categories. First, some additional functionality in the core code

was needed to handle TOPEX-specific issues (TOPEX file formats, handling of some special command types, etc). Secondly, a more robust and friendlier user interface was needed than the text-editor-dependent interface of the prototype. A graphical user interface based on X-windows is being developed, heavily re-using code that was developed for the rest of the TOPEX sequencing system. Thirdly, some convenience functions to assist people developing large sequences were added.

One of these “conveniences” was the addition of “intermediate carryover.” It is always necessary to carry over state information from the end of one sequence to the beginning of the next. Intermediate carryover, however, is used to allow MSAVE to start a run part-way into a sequence file, similar to the concept of “checkpointing” often used in large computational physics runs. If an anomaly occurs on the spacecraft, this capability can also be used to allow the user to change a state at a given time without a command (should the spacecraft exercise some of its automated fault protection, for example.) This same capability could also be used for “what-if” tests: if a sequence is built and someone in real time wants to know “what happens if I turned off this piece of hardware here with a realtime command”?

4.3. Testing issues

In addition, several issues arose that were not as critical in the research prototyping environment. Primarily these issues were in the realm of compatibility with existing software, testing and verification. The most significant of these was that the MSAVE system is novel among flight sequence checking software in that it allows a user to compile new rules and models into in-line code, thereby reducing the coding complexity and runtime penalties inherent in interpreted code while avoiding the inflexibility of hardcoded models. When a rule or model is added and saved, MSAVE will automatically recompile and relink the relevant MSAVE executables. However, the following issue inevitably arises:

“When we add a rule or model and add it to the code and recompile, what retesting is necessary?”

The addition of a rule or model is generating code, but the code is machine-generated. This machine-generation process will be exhaustively tested prior to project delivery. The syntax of the code is also

limited and predefined. Further, rules and models are isolated from the rest of the processing code, and a tracking scheme to avoid the possibility of overwriting variables or addresses in the rest of the executable has been developed. A syntax and format checking process takes place in MSAVE when a compilation request for a new rule or model is received to enforce this isolation. These limitations should bound the amount of testing that will be required.

A good analogy to the MSAVE testing situation is that when one develops flight code in C it is not seen as necessary to re-regression-test the C compiler for every delivery of the C software in addition to testing the code that has been written. The specification environment portion of MSAVE can be seen as a compiler for the flight rule and model language, and as such can be tested once and then the “programs” (the new rules and models as they are added) can be debugged separately. However, there is a capability in MSAVE to add functions in action tables for complex side-effect calculations. These functions will need to be generated in an editor, and limited functional testing of the system will be performed should a function like this be linked into MSAVE.

4.4. Limitations of the Verification Environment

MSAVE will not be able to detect constraint violations which occur after the end of a sequence, although if a subsequent sequence is checked they will be detected at that time. MSAVE verifies constraints using information available in the command sequence file only. This implies that certain constraints, such as those requiring precision modeling of spacecraft turns, solar array slewing, or orbit position propagation cannot be verified without significantly expanding the complexity and scope of rules and models in this implementation of MSAVE (although the general SAVE methodology in Section 2 could support this functionality).

4.5. Limitations of the Specification Environment

4.5.1. Command Scheduling And Cyclic Graphs

Sometimes it arises that a command has an effect later in time than the command itself; for example, a heater takes a thruster to the “warm” state after some time delay. These effects are modeled by “scheduling” a pseudocommand at a later (or, in some cases, the same) time. This ability to schedule commands is also the essential mechanism by which a given model

can effect changes upon the state of another model. The danger arises from self-scheduling commands, which can lead to a cyclic graph; i.e., Command A which schedules Command B which schedules Command A which whenever Command A or B is in the sequence file. This problem is handled in the flight version by prohibiting these loops, but a robust way of handling this situation is of interest for future versions.

4.5.2. Model Modification And Consistency Maintenance

The user will load into memory and edit only one model at any given time. Inconsistencies can arise because of this; for example, if Model B reads a state variable of Model A, and the user decides to delete this state variable in Model A, then Model B now has a reference to an undefined state variable. When the models are compiled, the inconsistency is detected and an error generated. It would be preferable to catch the error as soon as possible, but if MSAVE tried to catch all inconsistencies as each change to a model was entered, its performance would probably be unacceptable to the user if there were more than a few models in the system.

4.6. Experience Encoding Rules

We have now used the rules/action table system to develop and encode a variety of TOPEX flight rules. We have found the system to be a very good and systematic way of doing "knowledge capture" while the spacecraft experts who actually built the hardware are still around, since there is a tendency for a project to slowly lose expertise over time.

The action table format forces one to think through what happens if a command is sent in any of the possible spacecraft states that apply to that command, leading to thorough system behavior specification. It also leads to easier review and discussion than discussion of code in a programming language.

5. CONCLUSIONS AND PLANS

The specification and verification environment described in this paper has been successfully implemented as a prototype on a parallel machine, and is being fully implemented for the TOPEX/Poseidon project on a single-processor workstation. The environment has proven useful in generating real rules and models. When the full TOPEX implementation and rule/action table set is

available, we will take this large set and a set of sequences and determine the parallel efficiency of this full implementation. This will give us a basis for extrapolation for other missions and applications.

6. ACKNOWLEDGEMENTS

The authors would like to thank the JPL Director's Discretionary Fund and the TOPEX/Poseidon project for their interest in and support of this work. We also would like to recognize the contributions to the TOPEX-specific implementation by JPL's Dennis Page, Carlos Carrion and Robert Gustavson. Dr. Arkady Kanevsky of Texas A&M University, Dr. Krishna Kavi of the University of Texas at Arlington, and Dr. James Peters of the University of Arkansas had many helpful discussions with the authors during their tours as NASA/ASEE Summer Faculty Fellows at JPL. Joe Spitale was a participant in the Caltech Summer Undergraduate Research Fellowship program while working on this project. The work described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

7. REFERENCES

1. Horvath, J.C. and Perry, L.P., "Hypercubes for Critical Spacecraft Command Verification." AIAA-90-5095, In *Proceedings of AIAA/NASA 2nd International Symposium on Space Information Systems*, September 17-19, 1990, Pasadena CA.
2. Horvath, J.C., Tang, T., Perry, L.P., Cole, R.C., Olster, D.B. and Zipse, J.E., "Hypercubes for Critical Space Flight Command Operations." In *Proceedings of The Fifth Distributed Memory Computing Conference*, Charleston, SC, April 8-12, 1990.
3. Alkalaj, L.J., "Towards a Specification Language and Programming Environment for Concurrent Constraint Validation of Spacecraft Commands." JPL Internal Report, July 1992.
4. Chandy, K.M., and Misra, J., "Asynchronous Distributed Simulation via a Sequence of Parallel Computations." *Communications of the ACM*, Vol 24, No. 11, April 1981.
5. Schneider, K.M., "TOPEX/Poseidon Project Post-Expanded Checker Software Specification Document." JPL Internal Report, October 1992.

AGENDA : A TASK ORGANIZER AND SCHEDULER

Isabelle FRATTER

Centre National d'Etudes Spatiales
Toulouse, France

N 9 4 - 2 3 8 7 6

ABSTRACT

AGENDA will be the main tool used in running the SPOT 4 Earth Observation Satellite's Operational Control Center. It will reduce the operator's work load and make his task easier.

AGENDA sets up the work plan for a day of operations, automatically puts the day's tasks into sequence and monitors their progress in real time. Monitoring is centralized, while the tasks can be run on different computers in the Center. Being informed of any problems encountered, the operator can intervene at any time while an activity is taking place.

To carry out the various functions, the operator has an advanced, efficient, ergonomic graphic interface based on X11 and OSF/MOTIF.

Since AGENDA is the heart of the Center, it has to satisfy several constraints that have been taken into account during the various development phases.

AGENDA is currently in its final development stages.

Key words: Automatic task sequencing, Scheduling, Monitoring, MMI (Man/Machine Interface), Ergonomics.

1. INTRODUCTION

The Operational Control Centers for Earth Observation Satellites currently in operation (SPOT 1 and SPOT 2) require numerous manual interventions by the operators, who are responsible for:

- the Satellite Control Center for all the real time monitoring and control operations of the satellite while it is within visibility of the receiving

stations,

- the Platform and Payload Management Center for all off-line operations: orbit management, configuration control of the satellite's hardware and software, technological monitoring of the satellite, drawing up of the work schedule for payloads on the basis of requests from the Mission Centers.

It was therefore decided to design software that could automate the running of the Center for future observation satellites and thus lighten the operator's work load. The notion of an AGENDA was conceived.

AGENDA will be the main tool used in running the SPOT 4 Earth Observation Satellite's Operational Control Center.

2. AGENDA'S FUNCTIONS

2.1 Purpose and scope

AGENDA enables the setting up of a day of operations, the automation of the sequencing of the day's tasks, and the real-time monitoring of their progress. Monitoring is centralized although the tasks can be run on different computers in the Center. To make this possible, AGENDA takes account of the distributed nature of the Center's computer architecture. AGENDA informs the operator of any problems and he can intervene to influence the progress of an activity at any time.

All AGENDA's automatic actions and the manual interventions by the operator are recorded in a central log which constitutes the Center's records.

2.2 Notion of work schedule

AGENDA controls the applications through procedures which may be located and run on any

computer it supervises. Thus, AGENDA sees only the procedures and pays no attention to the contents of the applications. To plan all the procedures to be executed during a day of operations, the notion of a work schedule has been introduced.

Daily activity is described in a work schedule defined relative to a reference date. In a given work schedule, all the dates are defined with respect to this reference.

A work schedule is broken down into task sequences, themselves made up of procedures. All task sequences can be run in parallel, the procedures within the same sequence being performed sequentially.

Earliest activation dates and latest completion dates may be assigned to procedures. Procedures may wait for resources being used by other procedures and may depend on completion of procedures from other sequences.

2.3 Description of the functions offered by AGENDA

2.3.1 Setting up the daily work plan

AGENDA helps the operator set up the daily work plan. This preparation may be done in advance.

The daily plan is built up from pre-prepared skeleton plans (including all the operations to be performed during a nominal day) and information provided by the decisions made by the satellite operations control team. Each day is divided up according to the predicted times when the satellite comes into the field of visibility of the receiving stations.

Standard day skeletons are used to describe nominal days, each skeleton corresponding to a specific sequence of procedures that the operator can reset in time depending on his needs. It is simply necessary to change the reference date to be able to use the same skeleton on another date.

The operator can build a new work schedule or modify an existing one. For this purpose, AGENDA makes available to the operator:

- the list of existing skeletons,
- the pass forecasts,
- the list of available procedures with their estimated duration and the computers on which they can be run,
- a set of functions enabling an existing work schedule skeleton to be included in the work schedule being built up, procedures or chains of tasks to be added or deleted, and activation conditions to be defined for the various procedures (earliest activation dates, latest completion dates, start depending on completion of one or more other procedures).

AGENDA helps the operator to build his work schedule:

- it completes the information (e.g. by automatically calculating the total time duration of the work schedule, and the real start and finish dates for procedures),
- it updates the information (automatic update of all dates when a procedure is inserted into a work schedule, for example),
- it ensures that the information is consistent at all times (existence of procedures named, dates in increasing chronological order, condition circuits not leading to deadlock, etc.).

Once the work schedule has been built up, the operator can request its execution.

2.3.2 Monitoring the running of a work schedule

When the operator requests the running of a daily activity plan, AGENDA automatically triggers the automatic sequencing of the various tasks planned and monitors their progress in real time. It saves the run-time context and progress status at all times. Monitoring is centralized although the tasks can be run on different computers in the Center. The operator can intervene at any time while this activity is taking place.

All task chains run automatically in parallel. The procedures within a chain run sequentially. A procedure will be started by AGENDA if the following activation conditions are satisfied:

- the preceding procedure in the same task chain has been completed without error,
- its earliest execution time has been reached or passed,
- all the procedures conditioning its execution

have been completed without error.

The procedures inform AGENDA of their activity status through a run report. AGENDA then informs the operator.

A procedure may be:

- not yet started, because its activation date has not been reached,
- active,
- waiting for resources used by another procedure,
- stopped by the operator while running,
- ignored (the operator has specified that it should not be executed),
- correctly completed,
- stopped with error.

AGENDA signals an anomaly if the latest finish time for a procedure has been reached or exceeded but does not interrupt the automatic task sequencing. If an error occurs in a procedure, AGENDA displays it and stops the automatic sequencing of the corresponding task chain. The other chains continue unaffected. AGENDA warns the operator of anomalies or errors but does not take corrective decisions. These are the responsibility of the operator.

The operator can intervene manually on the automatic task sequencing. The options offered by AGENDA allow him to:

- interrupt a procedure that is running,
- delete an activation condition on a procedure,
- stop and restart a task chain,
- change the progress of a task chain by deciding to rerun or ignore certain procedures, thus jumping backwards or forwards,
- suspend execution of the work schedule. In this case the operator can restart execution where it was interrupted or modify the work schedule before restarting.

2.3.3 Other AGENDA functions

AGENDA also allows the operator to:

- record a message in the log to comment on the day of operations,
- start procedures manually, independently of the work schedule in progress and, if necessary, stop them during execution. He can thus perform a save operation that was not provided for in the daily plan, for example.

3. MAN/MACHINE INTERFACE (MMI)

To perform these various functions, AGENDA offers the operator an advanced, efficient, ergonomic graphic interface based on X11 and OSF/MOTIF.

3.1 Presentation of information

AGENDA gives an overview of the whole work schedule (past, present and future activity) in a single window. The information displayed is refreshed in real time. Information is shown on the screen in the same way whether the operator is preparing a work schedule or checking its execution.

Information is presented in graphic form and displayed versus time. Colors show the different activity statuses. This enables the activity of the whole Center to be assessed at a glance.

For example, a procedure is represented by a rectangle whose length depends on the time estimate, whose position on the screen is relative to planned activation date and whose color represents the activity status of the procedure.

All the following are shown in AGENDA's main window:

- name, start date, total estimated duration, and activity status of the work schedule,
- each task chain with its name, activity status, and the time left until the end of execution of the procedure in progress,
- each procedure, with its name, activity status, estimated duration and an indication of whether it depends on one or more other procedures,
- the satellite passes over the reception stations with the durations of visibility and the station names.

In addition, during running, the current time is given as a reference to situate the progress of the work schedule.

Figure 1 shows how the various information is presented to the operator.

The log messages concerning activity progression and operator actions are displayed in a scrolled window located above AGENDA's window on the same screen.

The operator can act on the portion of work schedule displayed by:

- making changes to the display scale (corresponding to zoom/unzoom),
- traveling in time (i.e. horizontally) using the arrow buttons in order to see past or future activity,
- moving vertically using the scrollbars to see other task chains.

3.2 Operator/AGENDA dialog

The operator controls AGENDA through the buttons, pop-up and pull-down menus, by

choosing from the scrolled lists, dialog boxes and other components of the advanced, user-friendly operator interface. AGENDA guides him at all times.

A specialized WYSIWYG graphic editor enables the operator to build up work schedules easily, and directly assess the result on the screen.

The operator can act directly on the chains and procedures or obtain information on passes by clicking on the corresponding graphic objects. He then obtains pop-up menus showing the various possible actions.

Figure 1. AGENDA: Example of presentation of information on the screen.

Piloter Programme de Travail		Editer Programme de Travail		Procedure		Journal de Bord		Session	
CCS-1 Active	epCommande Fin_Ck	Fin_Ck	Fin_Ck	Fin_Ck	Fin_Ck				
CCS-1 Active						TRANSFERT Fin_Ck			
STS-1 Active							ARCH-FACS Active	POURBOMBS Non_encore_lancee	
STS-2 Active								POUECOUP Non_encore_lancee	
DUCH-1 En_Alarme							TRAICRE Erreur	TRAICREP Non_encore_lancee	
OKMS-1 Active							TRAISMESURES Active	TRAIRACIO Non_encore	TRAIDATATION Non_encore_lancee
AUS									
Debut fenetre 00:08:00 Duree fenetre : 1 h v Nom du PGT : Journee_type Date Reference : 18/11/82 Etat : En_cours Duree : 00 24:00:00 Repere : 00:00:00 Passage Cacher Initialiser left right									

4. DEVELOPMENT CONCEPTS

4.1 Environment

AGENDA was designed for the SPOT 4 satellite's Operational Control Center, which consists of several HP computers operating under UNIX, together with X11 terminals connected to an Ethernet network.

4.2 Use of de facto standards

AGENDA's Man/Machine interface is based on the OSF/MOTIF environment and the X window System Version 11.

An interface builder (HP Architect) was used to design the Man/Machine Interface.

The use of the de facto standards (UNIX, X11, OSF/MOTIF) found in industry makes the software portable and thus usable in other Operational Control Centers.

4.3 Development constraints

As we have seen, AGENDA is the operator's main tool, starting and monitoring all the Center applications. This led us to take several constraints into consideration which had a strong influence on AGENDA's design and its different development phases.

4.3.1 MMI ergonomics

Great care was taken with AGENDA's ergonomics.

Ergonomics specifications applicable to the whole Center were drawn up to ensure that the screens presented to the operator were of uniform design (e.g. concerning the presentation of information, error messages, use of colors, etc.). These specifications were based on the OSF/MOTIF style guide.

The principle is to present simple screens that are easy to interpret (a single window gives an overview of all Center's activity) and easy and quick to use. Therefore it has been decided to reduce the input to a minimum by proposing choices from lists or using graphic representations of objects, and use only one level of menus. The risk of error is limited by always asking for confirmation for "dangerous" actions. A mock-up was made of the AGENDA MMI

and the prototype was criticised by the future Center users.

4.3.2 Efficiency

A special effort was made so that the AGENDA software would be efficient. As it is just a tool used to start the "useful" applications of the Center, it must not perturb the execution of these tasks.

On the other hand, being the operator's main tool, it must provide excellent response times both for responses to operator actions and for display refreshment and real time graphics animation.

A mock-up was also made for this constraint so as to validate the implementation choices.

4.3.3 Security

It is vital that correct execution of the daily planning be ensured at all times.

The "sensitive" functions (work schedule preparation and control of their execution) are accessible from a single work station under operator responsibility, whereas activity display functions are accessible by the various users from all the stations of the Center. Operator actions are checked, e.g. by requests for confirmation. The consistency of the work plan drawn up by the operator is continually checked. Progression of the activities and operator actions are recorded in a centralized log, which constitutes the Center records. This log is kept in a data bank and can be consulted at any time.

4.3.4 Taking cases of degraded operation into consideration

If a work schedule stops before time, it must be possible to restart the activity at the point where it was interrupted. Therefore, the consistency of procedure statuses must be ensured, including in the case of failure of one of the computers supervised by AGENDA or in the case of absence or malfunction of the software resources needed by AGENDA. The execution context of a work schedule is saved at all times, which enables the activity to be taken over by a backup computer, for example. Should a procedure be stopped early, restart points defined within the procedure mean that processing does not have to restart from the beginning but only from the last

point that was passed correctly.

4.3.5 Robust software

Being the "backbone" of the Center, AGENDA has to be robust. As the software is complex, because of its functions, the distributed computer architecture and the constraints described above, we attached great importance to development quality and having very complete tests.

4.3.6 Maintainability

The capacity of the Center was set to exploit SPOT 4 and its successor, SPOT 5. The software will be run for about 10 years, whence the importance of aptitude for corrective maintenance.

Furthermore, evolutive maintenance should also be possible and easy so as to be able to respond quickly and cheaply to future needs, for example to make an English version of the MMI.

The maintenance criterion is satisfied through quality development and the design and development methods chosen: separation of the "motor" and MMI codes, use of an interface builder allowing the MMI to be altered easily without touching the software "motor".

5. AGENDA TODAY AND TOMORROW

AGENDA is currently in its final stages of development.

A preliminary version is already available and is presently being used to prepare the skeletons that will serve for acceptance testing and the integration phases for the various subsystems composing the Center.

The final version will be available in May 1993. It will be used from 1993 onwards for the SPOT 4 Center's operational qualification phases.

Later, it will be possible to re-use AGENDA for other projects having similar needs in terms of daily activity planning and automatic centralized monitoring of multiple tasks.

COLUMBUS SYSTEM SUPPORT FOR TELESCEIENCE OPERATIONS

David W. Lytton[†]
 Rolf Schulze[‡]

N 9 4 - 2 3 8 7 7

[†] Space Applications Services (SAS)
 Place Flagey 7, B-1050, Brussels, Belgium
[‡] European Space Agency (ESTEC)
 Postbus 299, 2200 AG, Noordwijk, Netherlands

ABSTRACT

With the given constraints of the space environment, the telescience concept aims at providing a space mission user with optimum flexibility and responsiveness for space-borne investigations. The concept includes automated system management functions, which allocate and monitor planned resources and time windows, within which the investigator can perform his science interactively responding "on-line" to experimental data. During the telescience operation, the user is given the capability to send telecommands to the payload from the User Home Base with transparency to the rest of the system. Any violation of the "booked" time and resources will be detected by the system, and reported back to the user for appropriate action. Ultimately, the system will react to maintain the integrity of the system and its payload. Upon completion of the telescience session, the system management function reverses the system configuration and deallocates resources automatically.

Keywords:

COLUMBUS, APM, Telescience, Principal Investigator, Payload Operations

1. INTRODUCTION

The ESA COLUMBUS Attached Pressurised Module (APM), the European contribution to the Space Station Freedom, provides for an inhabitable laboratory environment and is conceived to be operated automatically, from the ground, and by the on-board crew.

The operation of on-board payloads can be done:

- either through pre-programmed automatic operations, invoked by the Space Station master timeline (Onboard Short Term Plan - OSTP), and processed accordingly by the APM element manager;
- or through telescience operations, defined as "the fully interactive mode of science operations whereby investigations on the Space Station and COLUMBUS elements are carried out under the effective scientific control of the investigator

teams" {ref.1}. This involves payload user operations via telecommands, issuing payload internal commands, with complete transparency of the rest of the APM system.

Both types of operation are supervised by automatic management functions to set and monitor operations envelopes, within which scientists can undertake their scientific investigations. Telescience is used for shorter duration payload activities such as swapping samples in an automated material processing furnace, whereas payload operations via the OSTP are used for long duration payloads and are characterised by Principal Investigator monitoring.

This paper outlines the COLUMBUS approach towards telescience operations, and the operational procedures, functions and communication services required to support the concept.

2. COLUMBUS APM OVERVIEW

Following complete attachment to the SSF node, activation and checkout, the APM will be ready to support payload activities as required by the SSF mission operations schedule. The payload activity support provided by the APM will include the operation of payloads from the International Partners (NASA, ESA, and CSA). The APM system will be under the control of the APM System and Mission Management (SMM) Software, which will respond to messages and commands sent from the SSF Integrated System Executive (ISE), the crew and ground. APM on-orbit servicing and maintenance will be planned and provide for the full 30 year operational life. It will be serviced and maintained during 90 day (*tbc*) increments, enabling the resupply of consumables, the changing of International Standard Payload Racks (ISPRs) or of individual payload subunits within the rack, and hence supporting a diverse range of scientific operations in the life sciences, space biomedicine and material sciences.

3. COLUMBUS OPERATIONS CONCEPT

3.1 APM Automatic Command Execution

In order to increase operational flexibility, automated system level management functions are provided,

serving as a tool for a safe and consistent implementation of external commands (ground, crew). In the APM this function is performed by the SMM which includes executable software and associated system and mission data. The SMM is responsible for:

- command processing (checking and execution);
- resource provision and allocation;
- failure management (system level Failure Detection, Isolation & Recovery - FDIR);
- reporting APM system data to the ISE.

This enables the APM to be operated as a ground-based laboratory without the continuous commanding and monitoring that has characterised previous spacecraft missions. This reduces the impact of limitations of up/downlink capabilities, and loss of signal (LOS).

Pre-planned automatic on-board operations within the APM are nominally invoked by the OSTP, running within the Space Station Manned Base (SSMB). Commands that are applicable to the APM are sent to and processed by the SMM, and may comprise time-tagged *Action_Names* or *Operational Task_Names*. An Action is the lowest operational step by which the APM can be controlled and operated within the context of nominal operations. It is associated with a step change in resource allocation/deallocation, and a functional process to be performed utilising this resource. An Operational Task (OT) groups together Actions related to the same mission goal.

3.2 APM Resource Management

The resource management function of the SMM handles all aspects of on-board resource management, including:

- resource allocation checks;
- resource allocation data management;
- resource consumption monitoring;
- resource reporting and logging to SSMB/crew/ground.

For the APM, the following principal resources are provided and managed:

- power distribution;
- data management services;
- cooling (air and water);
- vacuum line allocation;
- venting line allocation;
- nitrogen (purge gas) distribution to payloads.

Other resources are more effectively managed by the on-ground planning process (e.g. crew time, data relay satellite utilisation, microgravity environment), or by the SSMB (e.g. power provision, waste water removal, potable water supply).

Resource management is based upon operational resource envelopes. These are distributed amongst the individual users, and may be summed at either User Home Base (UHB) or at the collective User Support and Operations Centres (USOC) level. This is shown in figure 1. The sum of all the individual resource envelope allocations (e.g. USOC_a plus USOC_b) makes up the envelope for utilisation (i.e. the total amount of resources allocated to APM users).

Against these resources are set "soft" (operational) limits which are the limitations imposed upon the user of the particular resource, and "hard" (design) limits which represent the fixed thresholds of the resource, the exceeding of which may result in damage to the consumer.

4.0 TELESCIENCE OPERATIONS

4.1 Telescience Concept

The fundamental idea of telescience is to exploit the advances in both communications, and information technology to enhance the scientific return from space-based activities; the former are related to the exchange of information, the latter related to on-board data processing such as data reduction and ground-based user workstation software support. This forms the basis for a wide range of telescience applications, enabling the Principal Investigator (PI) to conduct scientific investigations in a decentralised manner (i.e. not in the SSCC or Payload Operations & Integration Centre - POIC) from a facility applicable to his particular scientific discipline, as shown in Figure 2. This may invoke the services of:

- quasi real-time/near real-time operations;
- teleoperations;
- telemanipulation (robotics);
- ground-based expert systems/AI.

The expected net benefits of this would include an optimisation of the utilisation of crew time, and an increase in the efficiency of the results obtained (both qualitative and quantitative) through the direct involvement of the PI in on-board payload operations.

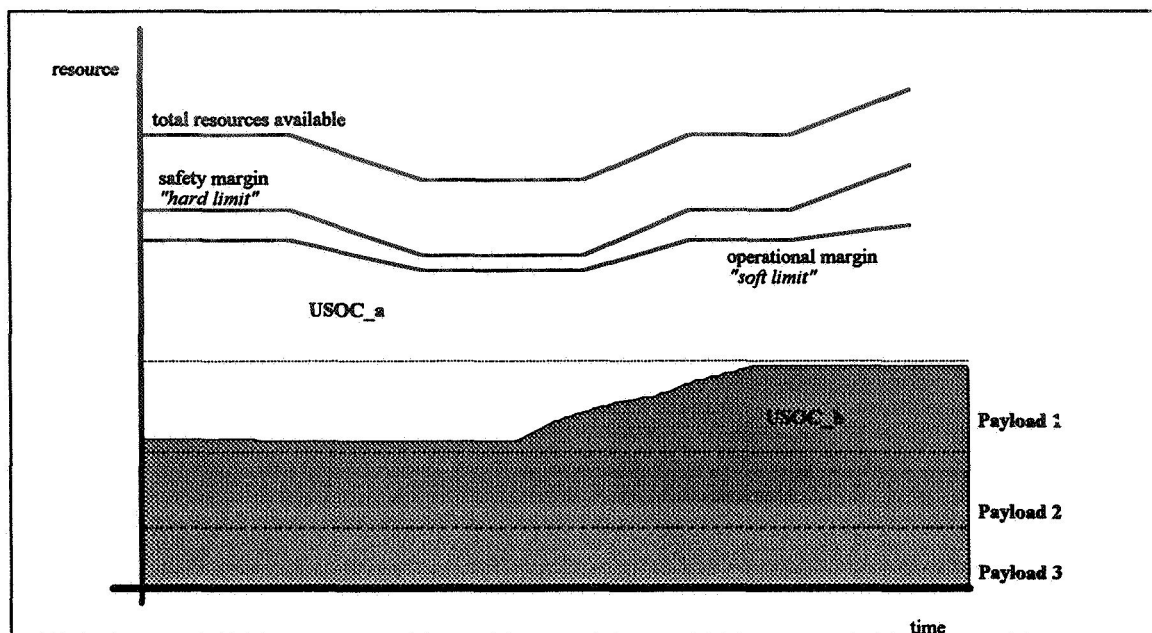


Figure 1 Resource Envelopes

A general telescience scheme, in which there is a mission operations control centre responsible for the monitoring and control of the flight system, and a set of PIs at their home bases is shown in figure 2. The users would receive the scientific data applicable to their particular experiment, whilst having the capability to directly send telescience commands to their experiments. This is performed with complete

transparency to the rest of the ground infrastructure, comprising the mission control centre for the system and payloads, and the ground stations providing the up/downlink. The expected duration of a telescience session is in the order of 15 minutes to 8 hours {ref. 1}, dependent upon the scientific discipline under investigation.

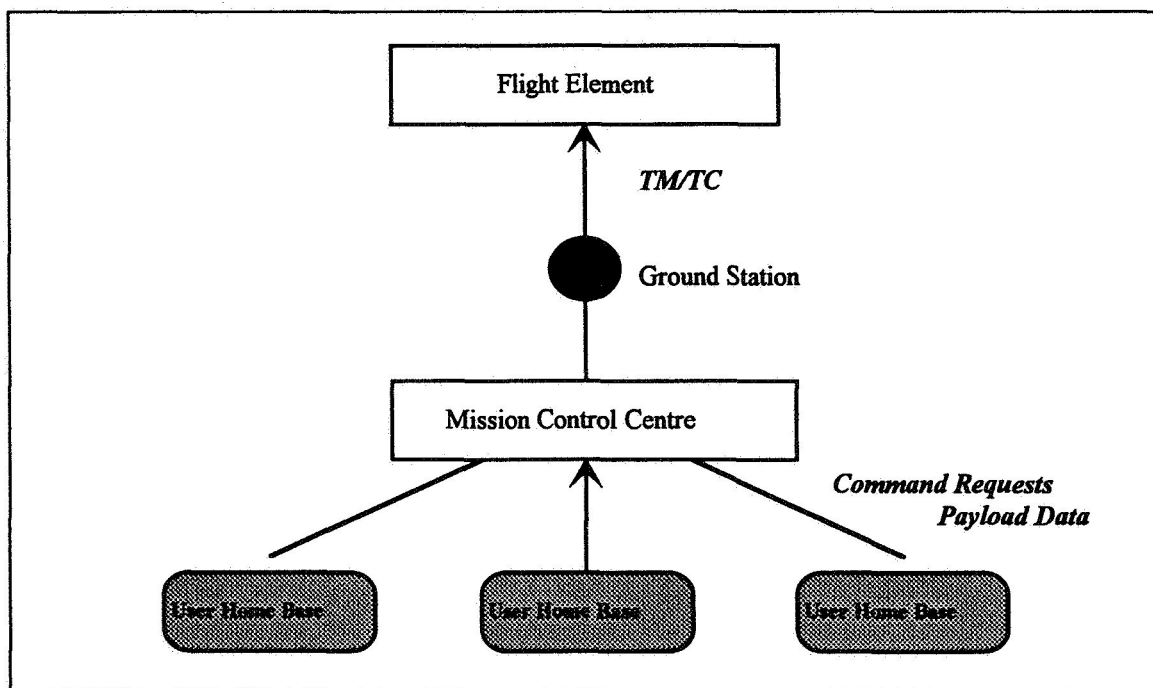


Figure 2 Basic Telescience Concept

4.2 APM Telescience Operations

4.2.1 Set-up of reference configuration and telescience mode

The operation of a particular payload in telescience mode will be identified in the OSTP, which will result in the set-up of a 'telescience mode' for the payload concerned, and its associated reference configuration. Attached to this will be the resource allocations for its particular operational period or window. This forms the boundary or envelope for the payload to operate within. The telescience period may run in parallel with the automatic operation of other payloads, defined within the OSTP. At the beginning of the operational window, as directed by the Actions and OTs, there will be an automatic configuration of the APM to support the payload operation, and the PI will be informed, when the system is configured for commencing telescience operations. The telescience operations themselves however, are left with the user. At the end of the operational period, the payload will be configured by telescience commands or the OSTP to a configurational status such that it is left in a stable state, and the APM system will be configured by the OSTP according to the next set of pre-planned operations.

4.2.2 Command authorised system resources for payload execution

As with the automatic operations mode, the APM system will be responsible for commanding those core system resources required to support payload execution (e.g. power, DMS services). The resources agreed by the ground will form operational and design envelopes which cannot be transgressed, but within which the PI will be able to operate, utilising those resource envelopes in real time. Transgression of these limits, including "booked time" would result in a warning being issued by the system requesting correction back to within the agreed margins. Failure to perform this would result in the system reacting to maintain the integrity of the system and the other payloads. The system will also initiate predefined responses if a system resource cannot be provided.

For telescience operations, resource management will be performed in two ways:

- *pro-active* checking, performed before execution;
- *reactive* checking performed on-board, but also on the ground (@ POIC/USOC) in real-time.

It is obviously desirable to eliminate as much redundant checking as possible, since this will

impose additional time delays upon the end-to-end data flow.

4.2.3 Direct provision to payload of payload-specific commands

The required types of direct activity are expected to include:

- transmission of on/off commands;
- setting of variable experiment parameters
 - furnace temperature,
 - time durations,
 - rotation angles;
- setting of on-board payload software control parameters;
- payload re-programming;
- continuous control (e.g. syringe control).

Commands and messages will be routed directly to the payload, without any intervention or checking by the SMM, which simply monitors resource consumption. Telescience commands comprise of 'real data' which are unintelligible to the system. They may form parametric values which are interpreted directly by the payload or by a payload-provided command sequencer. In the case of the latter, these sequences may issue internal commands held on-board, with interactive control by the PI (e.g. insertion of parametric values to the stored payload commands).

4.2.4 Optimised Onboard Data Routing

For telescience operations, optimised data routing with a minimum end-to-end communications delay is a pre-requisite for both uplink as well as downlink. From the user community's point of view, a forward link time delay of 2 seconds maximum and a return delay of 1 second maximum are preferable. At present, this is not supported, with a total end-to-end loop delay in the order of 10 seconds {ref. 2}. Ground links are considered in section 5.

The current onboard forward link response is dependent upon two factors:

- the performance of the onboard routers and associated protocols;
- the number of database accesses which any of the command processing elements will have to make (SSMB RunTime Object Database - RODB, SMM Onboard Database - OBDB), and the corresponding LAN loads.

An outline of the onboard routing (according to the architecture presented at COLUMBUS System Requirements Review (SRR) in July 1992), is shown in Figure 3:

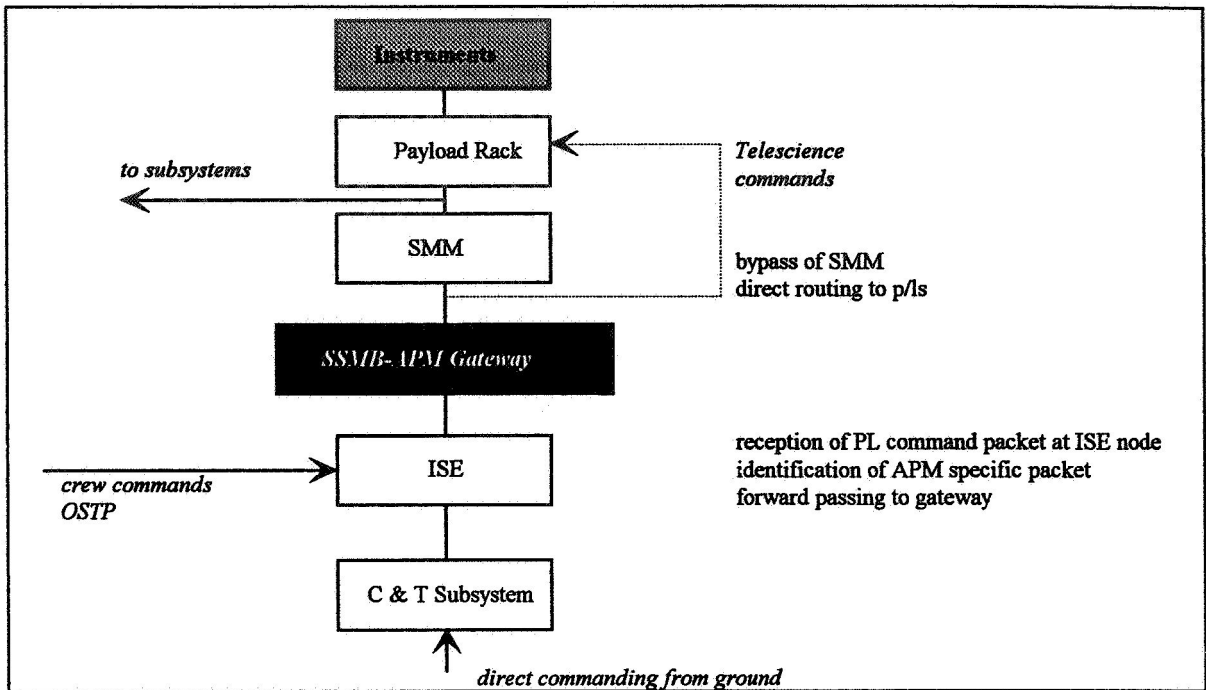


Figure 3 Telescience Command Routing

4.2.5 Overall Telescience Operations Concept

A general schema for the operation of the payload is shown in figure 4.

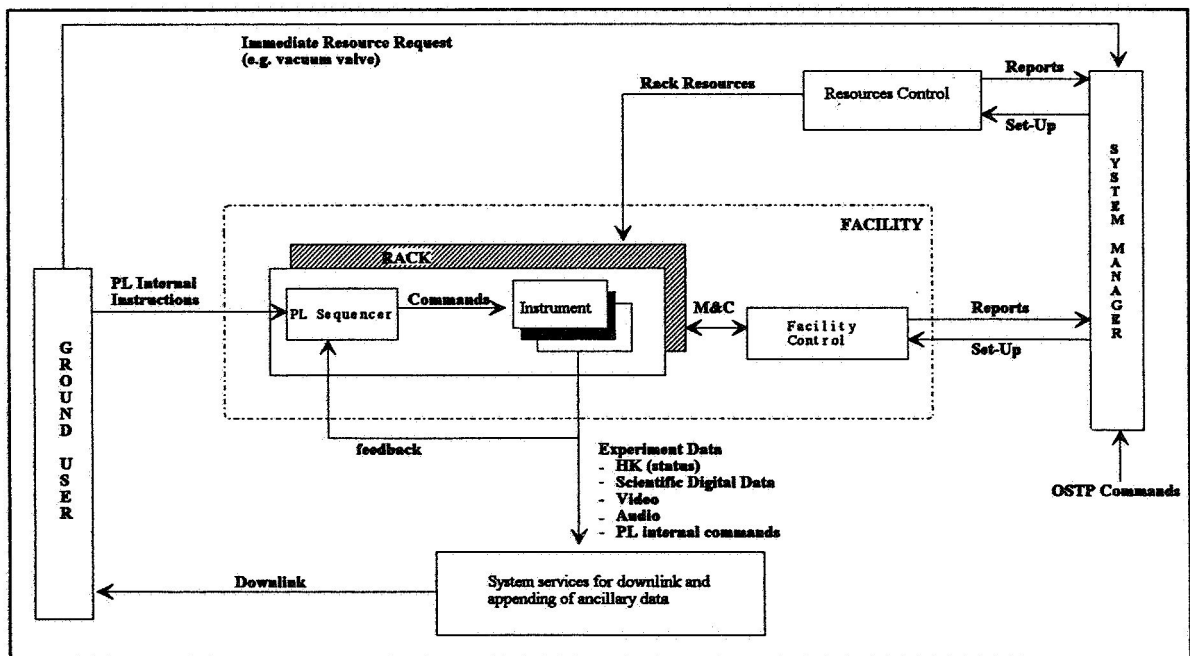


Figure 4 Telescience On-Board Operations Concept

5. SUPPORTING INFRA-STRUCTURE

5.1 Overview

The supporting European ground infrastructure for telescience operations is based upon a network connecting UHBs, USOCs and the Manned Space Laboratories Control Centre (MSCC). The UHB will be responsible for the health and use of its particular instrument, whilst the USOC is responsible for the payload facility in which the instrument may be contained. The MSCC (*tbc*) coordinates all European users of the SSF, and hence has to ensure that the resource budget that is allocated to the APM is adhered to, and coordinates the UHBs/USOCs accordingly.

5.2 User Home Bases (UHBs)

The UHB will have the following responsibilities:

- for a single experiment;
- execution of experiment;
- evaluation of scientific results;
- off-line planning interaction.

5.3 User Support and Operations Centres (USOCs)

A particular USOC will have the following functions:

- responsible for an onboard facility, containing a set of payload subunits;
- facility operations (including all system resources);
- facility checkout;
- facility support.

5.4 Operations Scenario

Interactive payload operations are based upon the ability of the user within the UHB/USOC to utilise an authorised set of commands which are routed through the POIC/CCC in the USA for authentication and authorization. The command set would be accessed via a user Work Station (UWS), allowing the control of the payload. These commands would have been validated off-line within an integrated simulation of the APM and would be checked on-line only for their source/destination authority.

Return data, comprising of payload housekeeping data and scientific data (up to 30.85 Mbps) is routed from the ground station directly to the USOC/UHB concerned.

6. CONCLUSIONS

Current telescience concepts have evolved around the use of manned and non-manned laboratories to be used primarily for microgravity disciplines in the Space Station context. However, current on-board design limitations and lack of continuous coverage inducing data relay satellite switchover problems impact end-to-end transparency to the user, as well as end-to-end commanding delays. This is compounded by discrepancies between national ground communication networks characteristics and mode of implementation, resulting in discrepancies for different users in terms of access to telescience operations, quality of service, and data capacity.

ACRONYMS

AI	Artificial Intelligence
APM	Attached Pressurised Module
CCC	Control Centre Complex
C&T	Comms & Tracking Subsystem
CSA	Canadian Space Agency
DMS	Data Management Subsystem
ESA	European Space Agency
FDIR	Failure Detection, Isolation & Recovery
ISE	Integrated Station Executive
ISPR	International Standard Payload Rack
LAN	Local Area Network
LOS	Loss of Signal
MMC	Mission Management Computer
MSCC	Manned Space Laboratories Control Centre
OBDB	Onboard Database
ODF	Operations Data File
OSTP	On-Board Short Term Plan
OT	Operational Task
PI	Principal Investigator
PL	Payload
POIC	Payload Operations & Integration Centre
RODB	Run Time Object Data Base
SMM	System and Mission Management
SRR	System Requirements Review
SSCC	Space Station Control Centre
SSF	Space Station Freedom
SSMB	Space Station Manned Base
UHB	User Home Base
USOC	User Support and Operations Centre

REFERENCES

1. Telescience Working Group Results, May 14-15, 1991
2. Columbus Users End-to-End Telematic Network (*COUETTE*) Final Report, July 1991

GENERIC MISSION PLANNING CONCEPTS FOR SPACE ASTRONOMY MISSIONS

N94-23878

O. T. Guffin and J. F. Onken

National Aeronautics and Space Administration
Marshall Space Flight Center, AL, U.S.A.

ABSTRACT

The past two decades have seen the rapid development of space astronomy, both manned and unmanned, and the concurrent proliferation of the operational concepts and software that have been produced to support each individual project. Having been involved in four of these missions since the '70's and three yet to fly in the present decade, the authors believe it is time to step back and evaluate this body of experience from a macro-systems point of view to determine the potential for generic mission planning concepts that could be applied to future missions. This paper presents an organized evaluation of astronomy mission planning functions, functional flows, iteration cycles, replanning activities and the requirements that drive individual concepts to specific solutions. The conclusions drawn from this exercise are then used to propose a generic concept that could support multiple missions.

Key Words: Space astronomy, mission planning, generic functions, design drivers, operations.

1. INTRODUCTION

Recent trends toward subsystem commonality in astronomy spacecraft and the concurrent desire to drastically reduce ground system support costs set the stage for concomitant reductions in the resources required when developing new mission planning concepts. The time has come for a careful evaluation of astronomy missions flown during the last twenty years, including lessons learned, to determine the basic mission planning and scheduling functions and their commonality from mission to mission. A logical organization of the requirements for a generic astronomy mission planning system can and will materialize from this evaluation. The three

main areas to analyze are: 1) mission planning functions, 2) functional flow considerations, and 3) design drivers. After examining the three areas thoroughly, it becomes very simple to describe the generic system and its core elements. A well-defined and implemented generic system will pay dividends by being able to support multiple satellites and by centralizing software maintenance and revisions.

2. MISSION PLANNING FUNCTIONS

The first step in this analytical approach is to look at each mission planning system (past, present, and future) and dissect it into its component functions. This process was done without regard for how the functions are packaged in software programs. The main generic functions that resulted from this analysis are described below.

2.1 Observation and Engineering Request Processing

The nature of astronomy mission planning requires a great deal of flexibility in receiving and processing observation requests. The astronomer using the system must be able to easily submit, edit, and track requests until implementation. In like manner, engineering requests arising from spacecraft health and safety considerations must also be developed by flight operations teams and entered into the scheduling stream where they can be merged with science requirements. This function may also serve as an executive driver for the other functions.

2.2 Orbital Mechanics

Because orbital mechanics is the foundation on which an astronomy mission plan is built, the mission planner should have easy access to the

spacecraft ephemeris. The orbital mechanics function should be able, based on the ephemeris, to produce rise/set times for any point in the sky. Also, the mission planner will need to be able to calculate line-of-sight angles for the spacecraft and/or its pointings. These pointings are evaluated for compatibility with the spacecraft and instrument constraints. The orbital mechanics function must also be able to track other satellites, Sun, Moon, and planet positions, compute ground site and South Atlantic Anomaly (SAA) entry/exit times and earth occultation periods, and generate spacecraft ground tracks.

2.3 Guide Star Selection

Once a target is chosen, the spacecraft will need to be given some guide stars near the target to fine tune the pointing angles and lock on the correct target. This can be done by the mission planners in a very straight-forward fashion. Software that incorporates the pointing algorithm of the telescope and the very latest star catalogs can be used to facilitate this task. The software should allow the user to easily manipulate star tracker/aspect camera characteristics in case of a partial or complete sensor failure. Depending on mission-specific constraints, the guide star selection function may run independently or in response to the scheduling function.

2.4 Scheduling

With observation and engineering requests and orbit mechanics data as inputs, the scheduling function produces and verifies an integrated spacecraft and viewing schedule. Past experience suggests that this scheduler should be able to work in two modes, manual and automatic. For a generic system, scheduling constraints should be easy to create, edit, and decipher. The manual mode should employ graphics capabilities using a mouse to insert and move observations and other events throughout the schedule. The product of this function should be a mission schedule that is completely free of constraint violations.

2.5 Editing

Once a schedule is produced, the next task is to be able to edit it. This function is necessary be-

cause mission schedules are usually generated weeks before execution, making them vulnerable to contingencies and significant ephemeris prediction errors. The editor should employ extensive graphics using a mouse to drag events around on the screen. The editor will include ephemeris information and target position information to calculate rise/set times as needed. Once a target is inserted into the schedule, it can be selected to show information about it at that time in the schedule. Since the editing function actually changes the schedule, it must also have a comprehensive validation capability to assure that the results maintain the schedule's integrity.

2.6 Communications Planning

Communications scheduling depends greatly on the interaction among spacecraft attitude and trajectory and satellite/ground site availability. For unmanned missions, communication opportunities are determined 3-4 weeks ahead of time; however, on Space Shuttle missions, the schedule is more determined by spacecraft orientation rather than communication satellite availability. Either way, the mission planner must have a way of determining communication opportunities and be able to send contact requests to and receive schedules from the Space Network (SN) or the Deep Space Network (DSN).

2.7 Spacecraft Management

Following schedule development, the spacecraft subsystem responses to execute scheduled events must be defined. Good design practice in modern astronomy satellites makes it more feasible to separate this increasingly independent function from the scheduling process. Events planned in the timeline, such as maneuvers and target pointings, dictate corresponding support from spacecraft appendages and subsystems. Solar array and high gain antenna movement and gimbal positions as well as data storage and communication device activities are included in this function.

2.8 Flight Operations Team Support

The mission planning and scheduling process by its very nature produces much information that is useful to flight operations teams. Typically,

this information is displayed on a large screen in a control center and/or fed to TV monitors and workstations to provide a focus or reference for mission operations.

2.9 Command Management

Although not a mission planning function per se, command management is included here for completeness because in some projects it is combined with mission planning as a matter of convenience or as the result of organizational responsibilities. On Spacelab missions, it is not a part of the mission planning and scheduling system.

MISSION PLANNING FUNCTIONS
• Observation and Engineering Request Processing • Orbital Mechanics • Guide Star Selection • Scheduling • Editing • Communications Planning • Spacecraft Management • Flight Operations Team Support • Command Management (optional)

Table 1. Mission Planning Functions

3. FUNCTIONAL FLOW CONSIDERATIONS

3.1 Modularity/Flexibility

As a mission planning concept becomes more generic, its component functions must be more modular and flexible. This is particularly true of the orbit mechanics and scheduling functions, which must satisfy the biggest share of project requirements and constraints. Each astronomy mission flown in the past has had its own unique set of science observation requirements and spacecraft operations, and although there is a tendency now towards some commonality, new software and science instrument advances will in the future spawn new requirements and options unforeseen today.

To posture itself for this onslaught of unknowns, the generic system must strive for a clean separation of the mission planning functions described above. Mission unique features can be modeled as input rules, algorithms and logic options in a scheduler with a standard user interface. Constraints that arise late in project

development or due to flight contingencies can often be satisfied by clever manipulation of orbit mechanics data. Functional modularity pays extra dividends by allowing standalone analysis of special operational problems and by making it possible to easily reorder functional flows whenever mission experience or contingencies dictate a change in the mission planning functional flow process.

3.2 Flow Sequence Optimization

One of the most important considerations in mission planning concept design is the order in which individual functions are performed. Careful attention to the sequence of activities involved assures an efficient, smooth-flowing system and saves many hours of extra effort during mission operations. A poorly designed flow, on the other hand, will cause many unnecessary iterations, confusion and added risk of error. For example, the checking of target visibility constraints after an observation has been scheduled instead of before is an open invitation to an endless iteration loop. The authors have noticed two errors in mission planning task flow that are common to many past and existing systems. One of the most ubiquitous mistakes is mixing orbital mechanics constraint calculations with scheduling algorithms in a minute-by-minute "sneak up on the limit" procedure. This approach is very repetitious and wasteful of computational resources. It is far more efficient to analytically solve for constraints such as target rise/set times one time in the orbit mechanics function and feed this information to the scheduling function.

Another common mistake is the early computation of spacecraft support events and commands in the scheduling process, long before they are actually needed. Combining these three independent functions creates a large amount of data that normally must sit around for several weeks before execution, making it vulnerable to contingencies and last minute changes. Changes to such volumes of data are labor-intensive and prone to error.

This last point illustrates another strong consideration in flow sequence optimization. External planning cycles, such as TDRSS resource scheduling, often dictate the generation

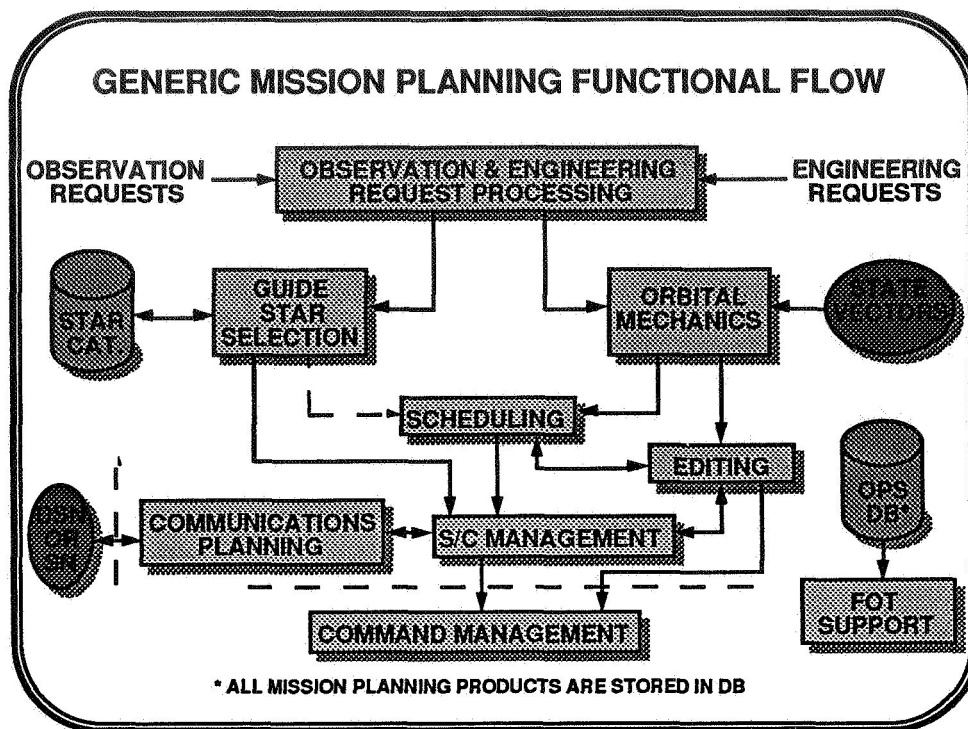


Figure 1. Generic Mission Planning Functional Flow

of observation schedules several weeks before they are required by a control center. This factor also leads to the conclusion that data should be generated when it is needed to minimize vulnerability and control center workload.

In summary, the two most important mission planning flow optimization principles are (1) perform the functions in the most efficient order to minimize iterations and (2) generate the mission planning products at the right time to minimize editing workload.

4. DESIGN DRIVERS

4.1 Science Instrument and Spacecraft Complexity

Fortunately, the major design drivers that affect astronomy mission planning concepts are relatively few (See Table 2). Of these, one of the most important is the complexity of science instrument and spacecraft design and the interaction between them. The pickle-shaped fields-of-view of the Hubble Space Telescope

(HST) Fine Guidance Sensors (FGS), for example, created a dependency among spacecraft attitude, guide-star selection and seasonal variations in target availability. Likewise, the dynamic rate of data production from its science instruments, ranging from micro-seconds to hours, sometimes makes it necessary to make a trial run through the command management system to assure that on-board memory and/or uplink restrictions will not be violated by projected observation schedules.

4.2 Real-time Requirements

Another closely related design driver is a requirement for real-time interaction with the spacecraft. This situation usually arises from the necessity to pinpoint faint object location and field orientations that are not well-known. To provide this capability, a link must be established between communication schedules done 3-4 weeks in advance and observation schedules. Last minute changes in planned contacts cause complex schedule interactions and require preplanned contingency procedures.

4.3 Manned vs. Unmanned

Compared with unmanned operations the advent of manned astronomy with the Astro-1 mission brought a new set of problems that significantly affect mission planning system design. Actually this design driver is composed of a number of different factors, but for purposes of this analysis they can all be attributed to having a "man in the loop." Two of these factors involved in Space Shuttle operations are the relatively short mission durations and frequent occurrences of launch delays, both of which dictate the independence of the orbital mechanics function, and the strong need for an efficient schedule editor and flexible scheduling software.

DESIGN DRIVERS
• Science Instrument and Spacecraft Complexity • Real-Time Requirements • Manned vs. Unmanned • Interfaces • Orbit Type

Table 2. Design Drivers

4.4 External Interfaces

Although rarely the result of any definitive requirement, the external interfaces with a mission planning system often have a significant effect on shaping its design. Organizational structure, roles and responsibilities, operational philosophies and the practicalities associated with shared facilities play an important part in concept development. These factors are normally assumed rather than being explicitly derived. For example, an organization responsible for both spacecraft and science operations will typically merge these independent functions together for convenience. Also, scheduling and command management are commonly combined if these responsibilities reside under a common directorate.

4.5 Orbit Types

The last, but definitely not least, design driver is orbit type. Because of the more rapid progression of environmental constraints, low earth orbit (LEO) missions are much more complex to plan than high earth orbit (HEO) missions.

Similarly, highly elliptic orbits present more ephemerides prediction and timing problems than do circular orbits.

5. CONCLUSIONS

5.1 The Generic System

After examining the component functions involved in a number of mission planning concepts, it appears that, if not one, at least a small number of generic systems are feasible. In fact, a first step towards a common concept might well be generic systems for certain types of missions. If this approach is taken, the authors recommend dividing the systems by orbit type and manned/unmanned operation. In any case, heavy emphasis should be placed on making the component functions as independent as possible, so that they can later be linked into one generic system. The generic system proposed by the authors is illustrated in Figure 1.

5.2 Core Elements

The key to making the transition from these interim generic systems to one system is the identification of "core" elements that change very little (or not at all) from mission to mission. Once these elements are standardized and used on several missions, the other elements can be examined one at a time to determine how they could be incorporated into the next-generation system. This close scrutiny will bring about new innovations in concept design and implementation, eventually leading to one generic system. Even if it doesn't make sense to provide a function that works for all missions, it is very possible to hook alternatives into the system. A prime example might be a different scheduler for manned and unmanned missions.

In like manner, functions and subfunctions not required for certain missions could be unplugged from the generic system. Spacelab missions, for example, don't need the spacecraft management element since the Shuttle Mission Control Center at Johnson Space Center performs this function.

In any case, all space astronomy missions share much common ground in the planning activities that must be performed to implement observation requirements. The time has come to exam-

ine these commonalities in search of a generic system to support future missions.

6. REFERENCES

1. Guffin, O. T. et al. 1985. "A Practical Scheduling Algorithm for Shuttle-Based Astronomy Missions." In AIAA 23rd Aerospace Sciences Meeting. Reno, Nevada: AIAA-85-0288.
2. Guffin, O.T. et al. 1992. "A Practical Approach to Astronomy Mission Replanning." In AIAA Space Programs and Technologies Conference. Huntsville, Alabama: AIAA 92-1472.
3. Olsen, C.D. et al. 1992. "Orbiter Attitude Design for the Astro-1 Spacelab Mission." In AIAA Space Programs and Technologies Conference. Huntsville, Alabama: AIAA 92-1466.
4. Sherrill, T.J. 1983. "Space Telescope Mission Planning." In AAS/AIAA Astrodynamics Specialist Conference. Lake Placid, New York: Paper 83-364.

SESSION E

MISSION CONTROL

CO-CHAIRPERSONS

M. Jones, ESA European Space Operations Centre, Germany
R. Moulder, JPL, California Institute of Technology, USA

Summary	299
M. Jones	
E.1 Mission Control Operations Concepts	
E.1.a A MOS for All Seasons	301
L. Bryant, JPL, California Institute of Technology, Pasadena, California, USA	
E.1.b An Operations Concept Methodology to Achieve Low-Cost Mission Operations	307
K. W. Ledbetter, NASA, Washington, D.C., USA; S. D. Wall, JPL, California Institute of Technology, Pasadena, California, USA	
E.1.c Mission Engineering	313
P. Ondrus, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA; M. Fatig, Bendix Field Engineering Corporation, Seabrook, Maryland, USA	
E.1.d Manned Space-Laboratories Control Center (MSCC) Operations Concept	319
J. Kehr, DLR German Space Operations Centre, Oberpfaffenhofen, Germany	
E.1.e Making Adaptable Systems Work for Mission Operations: A Case Study	327
B. E. Holder and M. E. Levesque, JPL, California Institute of Technology, Pasadena, California, USA	
E.1.f An Approach to the Design of Operations Systems	333
R. L. Chafin and P. S. Curran, JPL, California Institute of Technology, Pasadena, California, USA	
E.1.g Object-Oriented Technologies in a Multi-Mission Data System	343
S. C. Murphy, K. J. Miller, and J. J. Louie, JPL, California Institute of Technology, Pasadena, California, USA	
E.2 Mission Control Operations	
E.2.a Management of Information for Mission Operations Using Automated Keyword Referencing	349
R. A. Davidson and P. S. Curran, JPL, California Institute of Technology, Pasadena, California, USA	
E.2.b AMFESYS: Modelling and Diagnosis Functions for Operations Support	355
J. Wheadon, ESA European Space Operations Centre, Darmstadt, Germany	



E.2.c	Command and Control for the Israeli LEO Space Program	361
Y. Barnett, E. Gal-Ezer, G. Herman, A. Klein, and B. Walzer, Israeli Aircraft Industries/MBT, Yehud, Israel; O. Cohen and Z. Lenefsky, Decision Systems Israel		
E.2.d	Mars Observer Screen Display Design for a Multimission Environment	363
R. L. Chafin, JPL, California Institute of Technology, Pasadena, California, USA		
E.3	Mission Control Operations Tools	
E.3.a	The FOT Tool Kit Concept	371
M. Fatig, Bendix Field Engineering Corporation, Seabrook, Maryland, USA		
E.3.b	GSMS and Space Views: Advanced Spacecraft Monitoring Tools	375
D. Carlton and D. Vaules, Jr., Computer Sciences Corporation, Laurel, Maryland, USA; D. Mandl, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA		
E.3.c	VMPLOT: A Versatile Analysis Tool for Mission Operations	381
A. W. Bucher and O. G. Short, Martin Marietta Astronautics Group, Denver, Colorado, USA		
E.3.d	Real-Time Automated Failure Identification in the Control Center Complex (CCC)	387
S. Kirby, J. Lauritsen, and G. Pack, NASA Johnson Space Center, Houston, Texas, USA; A. Ha, S. Jowers, R. McNenny, and T. Truong, McDonnell Douglas Space Systems Company, Houston, Texas, USA; J. Dell, Loral Space Information Systems		
E.3.e	The Mission Events Graphic Generator Software: A Small Tool with Big Results	393
M. Lupisella, J. Leibee, and C. Scaffidi, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA		

SUMMARY — SESSION E

M. JONES

European Space Operations Centre

This session contained 15 papers covering various aspects of mission control, including general mission control problems, descriptions of some specific systems (both implemented and planned), and descriptions of tools for various aspects of mission control.

MISSION OR OPERATIONS ENGINEERING

The paper entitled “An Operations Concept Methodology to Achieve Low-Cost Mission Operations” by Ledbetter and Wall (E.1.b) focused on the important subject of operations concepts. The authors argued for preparation of an operations concept during the earliest stages of the project to help avoid the problem of a lack of coordination of spacecraft/mission design. They propose that the document change its purpose during the mission operations preparation, beginning as a design driver and ending up describing the final operations concept in the content of the actual design. This was felt to be an important paper, reflecting problems commonly found by operations system developers in all agencies.

“Mission Engineering” (E.1.c) and “FOT Tool Kit Concept” (E.3.a) dealt with the subject of the mission operations process, which is synonymous or overlaps with the terms mission engineering and total quality management. Low-cost missions are causing developers of mission operation systems to examine the whole development and operations process more closely, with a view to finding opportunities for cost saving through reuse, process improvement, and integration of the various operations aspects (spacecraft, science, ground segment, and space segment).

Another paper, “Management of Information for Mission Operations Using Automated Keyword Referencing” by Davidson and Curran (E.2.a) addressed the area of documentation. They noted that very large, costly bodies of documentation are developed from scratch for most missions, although they in fact normally contain much generic information. Moreover, these bound volumes are hard to maintain and lack cross-referencing. Thus, the process of mission information access needs improvement. The authors described a prototype tool using a hypertext user interface, which had been applied to existing mission documentation.

SPECIFIC SYSTEMS

Holder and Levesque presented a paper (E.1.e) on JPL’s multimission system for flight operations and control (SFOC), providing common data-processing facilities and tools for (in principle) all flight projects. A very interesting feature of this system is that it is based on heterogeneous workstations and uses open systems. A problem area raised in the question period was that of the influences of changes introduced for a new mission on already-running missions — this is indeed an issue for all multimission architectures.

MISSION CONTROL TECHNIQUES AND TOOLS

An interesting paper (E.1.f) was presented by Chafin and Curran on modeling of operations procedures using PDDL (a pseudo-English software design language). Such an approach could be a more effective way of writing, checking, and updating flight operations plans than the plain English descriptions conventionally used.

Another paper (E.2.d) by Chafin looked at more systematic but practical approaches to the design of displays, both fixed-format and flexible. Particular attention is paid to the grouping of information so that it is readily assimilated by operations staff. It would be interesting to see if general principles for screen design can be derived from this paper, since this is an area where arbitrary solutions tend to be taken for each project (or system).

Finally, a number of papers on graphical data presentation tools were presented. It should be noted that other such tools were presented elsewhere at the symposium (and also among the demonstrations). A major development has taken place in the area of data presentation, away from the traditional “busy” alphanumeric displays — only understandable after considerable training — towards much more user-friendly graphical displays. Space Views (presented by Carlton et al., paper E.3.b) is a good example of this, using 2D and 3D colour graphics to show spacecraft position and attitude relative to Earth, Sun, etc.; instrument field of view; and other items of interest to operation personnel. We expect such displays to rapidly become standard in spacecraft operations.

A MOS FOR ALL SEASONS

Larry Bryant

California Institute of Technology / Jet Propulsion Laboratory
Pasadena, California

N94-23879

ABSTRACT

From a systems perspective, this paper examines the challenges of a single system to support multiple Jet Propulsion Laboratory (JPL) space exploration missions and the need for unitary responsibility for the system. The focus is a Mission Operations System (MOS), which is effectively a mission management organization with direct authority over data system operations, command sequencing, flight operations control, data management, trajectory determination, telemetry and data acquisition, and spacecraft analysis. Stratagems for training and the approach to processes, procedures, and interfaces to facilitate the transition from the present situation to a truly multimission operational environment are developed.

The outcome is a paradigm for a MOS that is achievable, that can effectively support multiple projects, and that can take advantage of technological changes without perturbing the entire system.

Key Words: Mission Operations System, operations, customer focus

1. INTRODUCTION

In lean economic times, government expenditures receive close scrutiny. Certainly space programs where the benefit is not always intuitively obvious are subject to as close a look as any expense. One such project currently affected by cutbacks is Magellan, which is being terminated even though the spacecraft itself remains capable of operations for an extended period of time. This paper addresses one means by which a key factor in the Magellan demise may be mitigated for future projects, examines the Mission Operations System (MOS) for flight projects to see where change may be appropriate, and suggests a model for that potential change.

2. WHY CHANGE

It is clear from NASA's call for smaller, better, **CHEAPER** missions to explore space that costs are a major consideration in Jet Propulsion Laboratory (JPL) flight projects. The MOS is one prominent cost source that can be focused on for reduction. This is primarily due to the growth potential for these costs as we build better spacecraft that can and often must achieve longer missions. Magellan is a timely case in point. The projected costs of continuing the mission exceed the cost that can be justified by the anticipated return over the life of the spacecraft itself. Thus, endeavoring to reduce operational costs is an important aim in the continuation of space exploration. A contributing factor to the continuing accumulation of costs is the tendency of our interplanetary probes to substantially exceed their planned lifetimes. However, beyond the longevity of spacecraft, we find there are other significant operational cost drivers that can result in a relatively young mission, like Magellan, being phased out before the spacecraft is inoperable.

3. A SEASONAL MOS

An examination of the typical approach in establishing the MOS (or Ground System [GS], in Cassini's nomenclature) for a JPL flight project provides an insight into another cause of the significant growth of MOS costs. The MOS has traditionally been designed as a custom system to support a single project. The rationale for such an approach lies in the perception of the uniqueness of each project and its requirements for support. Further, we have only to look at the Voyager, Magellan, Ulysses, TOPEX, and Galileo projects to see that this approach has been successful in terms of mission performance. Each of these projects had or has a complete cadre of project-dedicated operational teams and support systems. Even though Magellan moved forward with the initial delivery of a multimission ground data system (MGDS), its version of the MGDS software is

unique and incompatible with the software versions of the other projects that are making the transition to the newer system. Even JPL's newest major project, Cassini, while using institutional or multimission resources that are available, still maintains a project-unique organizational structure for its ground system. It is time to look closer at this approach to determine the factors that spawn substantial costs, especially in comparison to the return they provide.

3.1 Let's Reinvent the Wheel

In the effort to design a MOS to support a specific project, there is a significant duplication of previous efforts and costs, that is, a real tendency to reinvent the wheel. The basic reason for the duplication is that there does not exist any single organization at JPL to take the lead and develop standards for operational processes, procedures, and interfaces that can be used directly or adapted for successive missions. An example within JPL is the matrix organization with expertise in MOS design, space flight operations, navigation, etc. spread among the technical divisions and then drawn upon and managed by each project as the project perceives the need. Even the Mission Operations System Office (MOSO), whose name sounds as if they might have a global operational role, has limited responsibilities in the areas of ground system development, mission control, and data management, to develop narrowly defined multimission capabilities. Consequently, projects like Cassini, rather than taking advantage of a standard operations approach through an established organization, continue to duplicate earlier efforts by designing their own MOS and expending the resources for boards to review the designs and the delivery schedules for project software, procedures, plans, etc. Another drawback to the current approach, which a single organization would mitigate, is the potential for under utilization of valuable resources.

3.2 Effective Personnel Utilization

Establishing project-dedicated teams, especially for nearly identical functions, can significantly under utilize a critical resource at JPL, namely, our people. The basis for this assertion is in the nature of mission operations that are continual but not continuous functions that must be executed in the conduct of operations. This means that during a normal workshift, a certain amount of dead time

can exist between real-time supports, while a software routine is running, or while data is being recorded for later analysis. This time could be used for support of other projects or tasks, assuming the person is properly qualified and trained. A classic example, and one in which some progress is being made, is mission control. It is not too hard to conceive of having a single individual to monitor a data playback for one project and a non-commanding tracking pass for a second project, either serially or in parallel, rather than having a dedicated controller for each project, as is currently the norm. The potential for unnecessary waste of an important resource clearly exists with our existing approach to mission operations systems.

4. CUSTOMER FOCUS

A single MOS would, in theory, have definite advantages, but we must assess how such a global approach would work in reality and how the development and transition should be approached to be certain of its ultimate success. To define what the role of a single MOS should be, we must deal with a fundamental issue not often dealt with directly at JPL. This issue is what Dr. Stone refers to as "Customer Focus" (Ref. 1). Who are the customers, what do they really need, and what is the right way to give them the right product the first time? In mission operations, we have customers who are both external and internal to JPL. Flight projects have external customers who are scientists and other users of the data from the instruments we send into space. Internally, the flight projects themselves are customers who expect a set of properly functioning instruments to be delivered to a specific locus in space, the instruments to be operated successfully, and the data from those instruments to be returned to Earth and accessible to the intended users. By focusing on what these customers want, we can identify an approach to mission operations that will in the long run be better and more cost-effective than our present system, that is, we can be "doing the right thing right" (Ref. 1).

4.1 The Customer's Real Requirements

The return of scientific data is the essential driver for mission operations. The ultimate goal of mission operations is to ensure that the data collected from space is provided error-free to the user; that is what the customer really wants.

Timeliness, correct format, time correlation, etc., are additional requirements that support the primary goal and that facilitate the use of the data. To satisfy its customer, a project places requirements on its own operations system. While projects have traditionally identified detailed requirements, like 24-hour tracking coverage during the first 30 days of flight or a complete checkout of every combination of ground system configurations, these are not the real requirements. Such requirements prevent the MOS from looking at options that may better satisfy the project's real requirements. Take the instance in which checkout and configuration of the spacecraft is accomplished within a matter of days, and the injection is so accurate that the first planned maneuver would be superfluous. The real requirements for getting the spacecraft safely into space and on the proper trajectory have been satisfied, and a real need for continued 24-hour coverage is questionable. It is apparent that the traditional project requirements on a MOS are not the truly essential requirements.

4.2 Pointing to a Single MOS

When requirements are couched in terms of functional performance, a different way of viewing how we put a MOS together begins to emerge. Rather than the project deciding that dedicated teams with specific staffing levels are required to support the traditional requirements, the project can focus on its needs in the context of functional performance and levy the responsibility for working the details on the MOS. Following this through, for a system to have the capacity, knowledge, capabilities, and skills to best conduct mission operations at JPL, it must access the experience and talent base of JPL across all operational disciplines. To do less would invite implementation of a less than optimal operational process for a project. For example, if the interplanetary navigation experience at JPL were ignored, a tremendous cost would be incurred to redevelop adequate algorithms to process tracking data and accurately determine a spacecraft trajectory. Thus, utilizing all disciplines is crucial and points to establishing a singularly managed MOS for all projects.

4.3 Moving Towards Cost Effectiveness

By melding the operational disciplines into a single entity responsible for delivering individual customer needs in terms of functional performance, a more

effective and efficient MOS can be achieved than is apparent in our current situation, in which several project MOSs compete for the best talent and resources. The reason is clear: sharing resources and talent provides each project access to the best concepts and an equitable share of support targeted at satisfying actual performance requirements. If, for instance, the planning and scheduling of tracking and commanding support is an integrated activity, then the supports can be distributed into a sequence of events that meets performance requirements and spreads the necessary workload around the clock. Unitary management of a MOS to support all projects leads to more effective and efficient operations and brings us to the point of addressing what the role and structure of the MOS should be.

5. A MOS FOR ALL SEASONS

The necessity of accessing a broad spectrum of talent and skills within JPL implies a broad role for the MOS to integrate mission operations across the range of JPL projects more efficiently and effectively. This role would appropriately cover all aspects of operations from the pre-launch tests and launch operations through the entire mission until delivery of all the captured data to the science community. Such a broad scope of coverage provides operational continuity during the entire opportunity for significant contact and interaction with the spacecraft under operational or quasi-operational conditions. Responsibilities of the MOS would encompass managing the delivery of ground data system capabilities that can support each mission and that provide an integrated capability so as not to adversely affect the ability or cost for the MOS to support all projects. The planning of operations and spacecraft sequencing in response to mission plans and endeavoring to reduce interference or conflicts between support requirements and distribute the workload to make staffing management easier is also within the purview of the MOS. Resource scheduling for both flight projects and external requirements would be a MOS task to provide a single clearinghouse that can make decisions to resolve potential conflicts. The direction and implementation of all aspects of operations, including operating the ground system, commanding the spacecraft, trajectory determination and prediction, telemetry processing, spacecraft health and status analysis, data archiving and distribution, with associated support processes, are clearly within the charter of this MOS.

Additionally, development of standard processes and procedures as well as operational interfaces to simplify and facilitate the capability of supporting all projects effectively, and the dissemination of these good operational practices through formal training are significant elements of the MOS role. This latter aspect of standardization and training is particularly important in achieving cost effectiveness across all projects and can be seen to be quite practical once one looks at what the functional performance requirements are for operations versus the current approach for specifying MOS requirements. Embracing such a broad scope for the MOS role is needed to ensure the efficiency and effectiveness of the MOS to satisfy the real operational requirements—that of safely delivering a set of instruments to space and returning the scientific data they collect in a usable form to the science community.

5.1 Looks Like . . .

The selected structure (see Figure 1) of the MOS must complement the breadth of its role. This implies an organizational level on a par with that of flight projects to facilitate communication and a cooperative effort to best achieve the primary goal of each project. To further facilitate effective cooperation and equitable treatment, a dedicated project mission manager is an invaluable member of the MOS staff. Other elements of a single MOS would parallel those of existing MOSs except that they would all be responsible for the full range of JPL flight projects. Some of these elements are either in place or being implemented in the areas of mission control, data system operations, and telemetry and data acquisition operations. Others, like data management and navigation, have the capability already and merely require the formal organizational realignment to achieve full multimission capability within a single element. The design and generation of command sequences would be handled by a single element, although there would surely be some pockets of expertise for any essential project-unique applications. Still, the basic concepts, especially as processes are standardized, are similar, if not identical, which makes support of distinct projects a practical matter. A planning and integration element is vital to the unified MOS. The need to distribute supports and coordinate and integrate activities to apportion workload and resource utilization equitably is fundamental and can only be achieved through a single element with the responsibility and

authority to develop operational plans and to allocate resources accordingly. Perhaps the most challenging domain for consolidation is the analysis of spacecraft health and status and contingency responses. This would, in fact, involve spacecraft dedicated effort during critical events. A team of spacecraft subsystem experts would be routinely involved in daily health and status, event verification, and command planning and review activities. Each individual would have received vehicle-specific training in more than one of our spacecraft to provide for backup during high activity and coverage during absences. Additionally, during and immediately following launch and until the spacecraft is safely configured and characterized, personnel with an in-depth knowledge of the spacecraft gained during development, test, and launch preparation activities would supplement the primary analysis element. Additional expertise would also be identified for support during selected critical activities as identified by a project's mission manager. The concept for this organizational structure is intended to ensure that each project's actual requirements can and are met in a safe and timely manner while utilizing available workpower and other resources effectively and efficiently. An essential element to achieve this efficiency is a structured training and certification effort that promulgates the standardization and implementation of good operational practices as well as addresses those areas of knowledge and capabilities that are really project-unique. The result of this organizational approach is a MOS that can satisfy the functional performance requirements of existing JPL projects and make the necessary adjustments or growth to do the same for future projects.

5.2 Not a Yellow Brick Road

To get to such a MOS organization is not an overnight task. It will require extensive planning and even cultural changes. Just as it is not an overnight task, neither is it a perpetual task. The approach is straightforward and doable if the commitment is forthcoming. The initial effort involves establishing a core organization with operational, project, and training expertise to lay the foundation. This foundation involves establishing standard approaches to operations, chronicling good operational practices, drafting interface control documents, and developing positional and scenario operational training. Once this foundation is developed and codified, the next

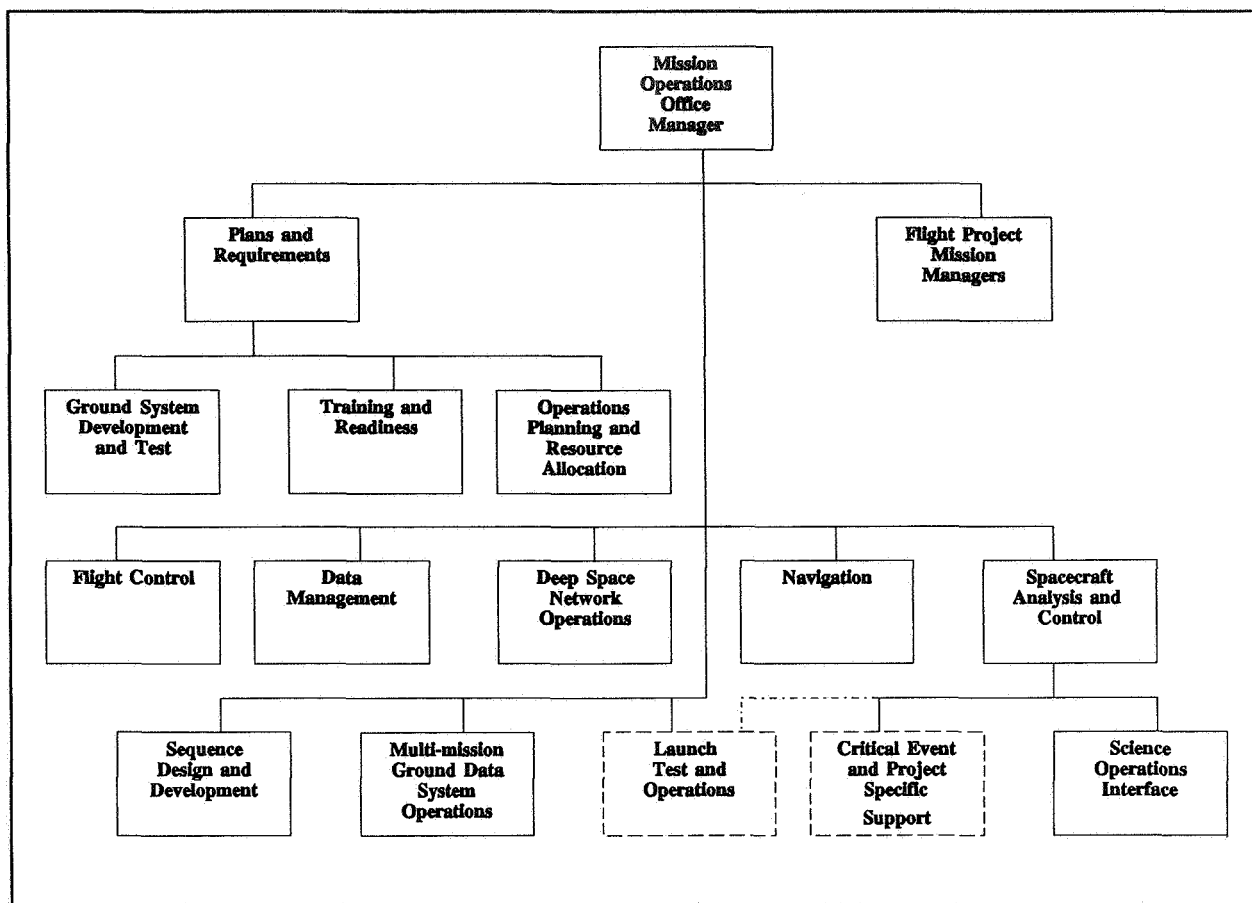


Figure 1 - Candidate MOS Structure

phase, the implementation of the training programs, begins. This activity prepares the existing project-dedicated teams to merge personnel and expand their scope of responsibilities. When this phase has been accomplished, the staffing can be adjusted to match the operational workload. Finally, the process is established for incorporation of new projects. This process includes development of training, delivery of the training, implementation of operations, and adjustment of staffing.

Another point needs to be made regarding the MOS and the implementation of standard processes, procedures, and interfaces. One might logically assume that taking such an approach and controlling these items could lead to stagnation within the MOS with respect to new techniques and technology. Such is not the case. In fact, such an approach actually facilitates change and improvement. First, through the control process, proposed changes and modifications are assessed as to their benefit and cost to the MOS as a whole. Then, because interfaces are controlled, it is possible to implement

a new technique within a subsystem or element or to apply a technological advance, such as tools utilizing expert systems, without disrupting the entire system. To the rest of the system, with proper interface control, a changed element or subsystem need not look any different. Improvements can be made and even interfaces changed if there is a proper cost-benefit ratio. The crucial factors are that the change is controlled and documented and that the effects of the change are incorporated into the operational training whenever appropriate.

5.3 A Cornerstone

Because effective training is fundamental to the success of the implementation of a single MOS, it is appropriate to consider the training approach to be implemented. The approach would be systematic and based upon the operational jobs to be performed. It would differ from other approaches to technical training in that the emphasis is on how the system functions (Ref. 2) and the

cognitive involvement of operations personnel as opposed to a structured set of tasks typically associated with technical training. In developing the training for the cognitive involvement, our operational training would also rely upon network analyses of operational and system experts' models (Ref. 3) of the spacecraft, the ground systems, and the operational processes. The results of these analyses would be used to structure both the positional and scenario training for operations with the content based on the job performance requirements. The training process also involves validation through MOS-level rehearsals and readiness testing as well as updating to improve effectiveness and incorporate new projects. By using this systematic approach, the training element can in fact effectively provide the foundation upon which to build a MOS for all seasons.

6. OPINION

With the need to hold the line on budgets, operations costs are a valid arena to examine, as demonstrated by the Magellan project. By addressing the real performance requirements, it is possible to identify a MOS structure that can be effective without being dedicated exclusively to a single flight project. The role of a single MOS would necessarily span all of operations, from pre-launch through mission operations to project termination. To be effective, the MOS would be a distinct organization with authority to make and implement the decisions that are the organization's responsibility. The foundation for such an organization is assured in the development of standard processes, procedures, and interfaces and, of utmost importance, in a formal training process to propagate these standards and good operational practices. It would also prepare the system for new projects as they develop. With the application of the necessary effort, JPL can achieve a single MOS management structure that draws from the matrix organization to support all flight projects cost effectively and to stand the projects in good stead whatever the season.

7. ACKNOWLEDGMENT

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

8. REFERENCES

1. Stone, Edward C., October 9, 1992. "Why does JPL need the TQM Initiative?" *Universe*, Jet Propulsion Laboratory.
2. Widdowson, J.M. and Taylor, D., 1992. "Training for Manned Spaceflight." In *Journal of the British Interplanetary Society*, Vol. 45: 77-80.
3. Ayers, T.J. and Bryant, L., 1989. "Training for Spacecraft Technical Analysts." In *Proceedings of the HFS, 33rd Annual Meeting*, 1263-1267.

AN OPERATIONS CONCEPT METHODOLOGY TO ACHIEVE LOW-COST MISSION OPERATIONS

N94-23880

Kenneth W. Ledbetter
NASA Headquarters
Office of Space Science and
Applications

Stephen D. Wall
Jet Propulsion Laboratory
California Institute of Technology

ABSTRACT

Historically, the Mission Operations System (MOS) for a space mission has been designed last because it is needed last. This has usually meant that the ground system must adjust to the flight vehicle design, sometimes at a significant cost. As newer missions have increasingly longer flight operations lifetimes, the MOS becomes proportionally more difficult and more resource-consuming. We can no longer afford to design the MOS last. The MOS concept may well drive the spacecraft, instrument and mission designs, as well as the ground system. This paper presents a method to help avoid these difficulties, responding to the changing nature of mission operations. It is this paper's thesis that proper development and use of an Operations Concept document results in a combined flight and ground system design yielding enhanced operability and producing increased flexibility for less cost.

Key Words: Operations concept, mission operations, mission control, operational requirements, program phases.

1. INTRODUCTION

For most missions, beginning the design of the Mission Operations System (MOS) occurs only after the spacecraft and mission design are well established. It is difficult for a project to consider operational issues that impact the program several years in the future when attention is focused on an immediate spacecraft design or fabrication problem. When early decisions impacting the ground system are made by spacecraft, instrument or mission designers without input from MOS designers, difficulties in MOS design and implementation arise. As a result, costly solutions are sometimes necessary. An example of this was the way telemetry from the Viking lander experiments was packaged

into different sized, independent data frames rather than packetized into a master frame. The seismometry frame was even variable in length. The signature of a marsquake could be adequately captured only by a high sampling rate, therefore the seismology data frame was designed with the unique capability to expand to a larger number of bytes if a quake were to trigger the instrument. This innovative idea may have served the experiment well during operations, but the design of the telemetry frame synchronization process on the ground was made unnecessarily complicated because of variable length data frames.

With recent concerted emphasis on reducing the cost of space programs, the contribution due to designing and building the MOS has to be considered along with that of designing and building the spacecraft. James S. Martin, NASA consultant and former Viking Project manager has said, "As more and more missions are planned to have flight operations lifetimes measured in years, even tens of years, the MOS becomes an increasingly more difficult, more costly, and more human resource limited process. No longer can we afford to design the MOS last. In fact, the MOS concept may well drive the spacecraft design, the science instrument design, and the ground system." (From the Foreword to Ref. 1)

There has been recent progress toward involving the MOS designers in the process of spacecraft and mission design from the beginning. This paper describes how a key document, the *Operations Concept*, can assist this process. Proper development and use of the Operations Concept can result in the designs of the combined flight and ground system yielding enhanced operability, thereby producing increased flexibility for less cost. Discussion of the Operations Concept is best made in the context of standard program phases, reviews

and documentation. However, since these vary between NASA centers, the following section will define the context for this paper.

2. PROGRAM PHASES AND REVIEWS.

The four phases of a program's life cycle are: (1) *system planning*, which comprises feasibility analysis, concept development, requirements definition, and the development of a system acquisition plan; (2) *system development*, which includes requirements analysis, concept evaluation, preliminary and detailed design, implementation, and test activities through acceptance testing; (3) *integration and test*, which encompasses the integration of systems into the overall MOS and complete testing of the integrated systems against all the specified requirements; and (4) *system operations*, which consists of flight operations support, beginning with launch operations.

In spite of the extra work they cause for the system designers, formal reviews of the progress of the project before a board of experts are essential to resultant design excellence. A series of reviews should be held, independently for the spacecraft and for the MOS. Figure 1 illustrates the timing of various reviews with respect to the program phases. During the System Planning phase, there should be a *Preliminary Requirements Review* (PRR) after the initial requirements definition is complete, to assess whether or not the requirements are well defined. Historically, more projects have gotten into serious difficulties due to incomplete or ambiguously defined requirements than any

other cause. A *Preliminary Concept Review* (PCR) is recommended after the full system operations concept is developed.

During the System Development phase, a *System Requirements Review* (SRR) and the resolution of action items that follow will finalize the requirements and evaluate readiness for design to begin. After preliminary system design is complete, a *Preliminary Design Review* (PDR) is held. The preliminary design should be approved before detailed design can begin. When the design is complete, the *Critical Design Review* (CDR) is the milestone preceding procurement of hardware and start of actual implementation. In practice, multiple CDRs are usually held for various components of the system, due to the additional detail and differing review board expertise. Unfortunately, many times the design begins before the requirements are finalized, the project bowing to time and schedule pressure. A well developed concept of operations will help definitize the requirements early. Minimizing the amount of design accomplished before finalizing requirements lowers the ultimate cost of the system by eliminating redesign.

During the Integration and Test phase, there are many reviews of specific activities to ensure the implementation and testing is proceeding according to plan. MOS components, consisting of both hardware and software, are incrementally delivered after each has completed its user acceptance testing. However, when the fully developed MOS ground system is ready to be delivered, an *Acceptance Test Review* (ATR) is held. The ATR reviews the testing against the original requirements and against the operations concept. Once all MOS subsystems are delivered and the flight operations team is selected and trained, the *Operations Readiness Review* (ORR) is held to verify that the entire MOS, including the people who will operate it, is ready for operations to begin.

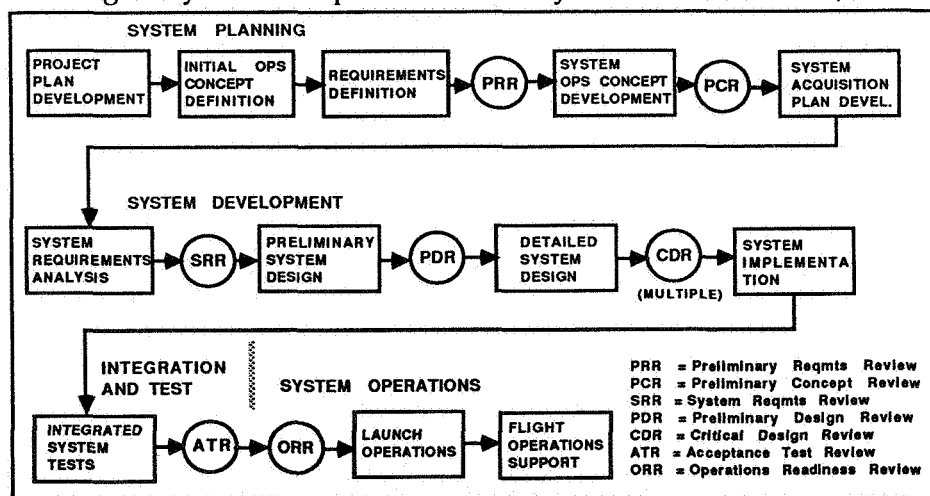


Figure 1 - Program Phases and Reviews

For small projects where costs are severely limited and the complexity of flight components and ground systems are not as great, the reviews may also be less expansive in scope. (As an example, the preliminary concept may be reviewed at the same time as the preliminary requirements, effectively combining the PRR and PCR.) Although some reviews may be combined, it is important not to omit the actual step from the process, for the efficiency of the resulting design depends on orderly execution of the design and implementation process.

3. OPERATIONS CONCEPT.

An Operations Concept is an orderly collection of user-oriented ideas as to how the operations system should function to satisfy the mission and experiment objectives. Although the concept can exist without being documented, a written version ensures uniform dissemination of the concept. The assumption here is that the Operations Concept is a summary document describing the collection of ideas that form the concept.

The Operations Concept is an evolving document, with a purpose and function that changes with the program phases. Its initial and most important function is that it (1) *drives the program design* toward one that will satisfy the mission and experiment objectives. Written early in the system planning phase, this initial version establishes and clarifies the intended operational approaches. As the design of the operations system is developed, the Operations Concept (2) *guides the design engineers* by shaping the definition of system requirements and keeping the focus on system operability. With additions and configuration-controlled revisions as necessary, the Operations Concept, at the end of the design phase, (3) *becomes a summary description of the design*, illustrating the way the operations system will be used to conduct mission operations. The objectives of the Operations Concept and its uses are indicated by the following nine items, each allocated to one of the above three document purposes.

As a design-driving document, the Ops Concept:

1. *Summarizes the primary objectives and constraints for both the mission and experiments.* These basic goals are established by the project scientists and mission planners and

become an input to the system planning phase.

2. *Documents early the intended operational approaches.* Basic operational philosophy delineated early in the planning phase will simplify subsequent tasks and establish design requirements.

3. *Defines how users will operate and maintain the system.* The domain of user activities is defined. System operations, system maintenance and required institutional support are all addressed early enough to influence the design requirements.

As a design-guiding document, the Ops Concept:

4. *Becomes the unifying document for the requirements analysis and design phases.* By clearly defining the operational use of the system, it serves as a reference for designers, communicating the operations strategies to project personnel as system definition and design proceed. Systems engineers will consult it for guidance to ensure the system design will satisfy the operator's requirements. It also provides a test bed where design issues can be raised and resolved.

5. *Clarifies operational interfaces early.* It identifies operations interfaces early enough in the program to ensure a common understanding and sufficient definition, resulting in a more efficient implementation. Interface identification also defines the environment for the integration test program by specifying which operational components must "play together."

6. *Provides a framework for trade and cost studies.* By defining and prioritizing necessary system operational features, the operations concept will provide criteria for evaluating trade study and cost options.

As a design-description document, the Operations Concept:

7. *Provides input for generation of plans and procedures.* It supplies information for various operations plans such as the Mission Data Plan (which describes the handling plan for downlinked information) and the Experiment Operations Plan (which explains how the experimenters will operate their instruments). It also provides input for generation of team operations plans and subsequent team activity procedures.

8. *Supplies test objectives for system integration tests.* This will ensure that the testing will prove the mission concept and that

the design will meet established requirements.

9. *Summarizes intended mission operations.* It always remains a concise, readable summary of the purpose and intended operation of the mission at any time during any phase, for either external or internal interests. It is required reading for new personnel and a valuable source of operations training material.

The initial version of the Operations Concept is written after only the rudimentary mission objectives have been determined, and in fact must be formed in parallel with those early mission concepts. As illustrated by Figure 2, it is generated from conceptual ideas of how the system will be operationally used to satisfy mission objectives. In most cases, this first version is written by the office funding the mission for attachment to requests for study proposals. Its purpose is to provide information from which a contracting company or university can proceed to develop science requirements, mission requirements and the operations design requirements that support them. At this stage, the contents are the objectives and constraints of both the mission and individual experiments, and the conceptual approaches to operations system activities. Once determined and agreed upon by the participants, the operational approaches that constitute this initial concept should be placed under configuration control. They cannot be

arbitrarily changed without the consent of all affected parties. This initial concept is a major source of operations design requirements generated as an input to the functional requirements documents.

Once the initial set of MOS design requirements are defined, the first version of a full Operations Concept document can be written and reviewed at the PCR. Figure 3 provides a sample outline for the content of this complete document, taken largely from Ref 2 with modifications. Although it is likely to have many incomplete sections for those areas where detail is dependent on a design selection, it will contain the basic concept for operating the system as it is initially envisioned. It will define at a high level the intended uplink process for planning, scheduling, generating, validating, and transmitting commands or sequences of commands, and the downlink process for receiving, monitoring, separating, and processing the telemetry. These strategies are not final at this stage but will evolve with the design.

In order to efficiently implement an Operations Concept, several ground rules need to be imposed during development. Adherence to these rules could make the difference between success and failure of the Operations Concept as a useful tool.

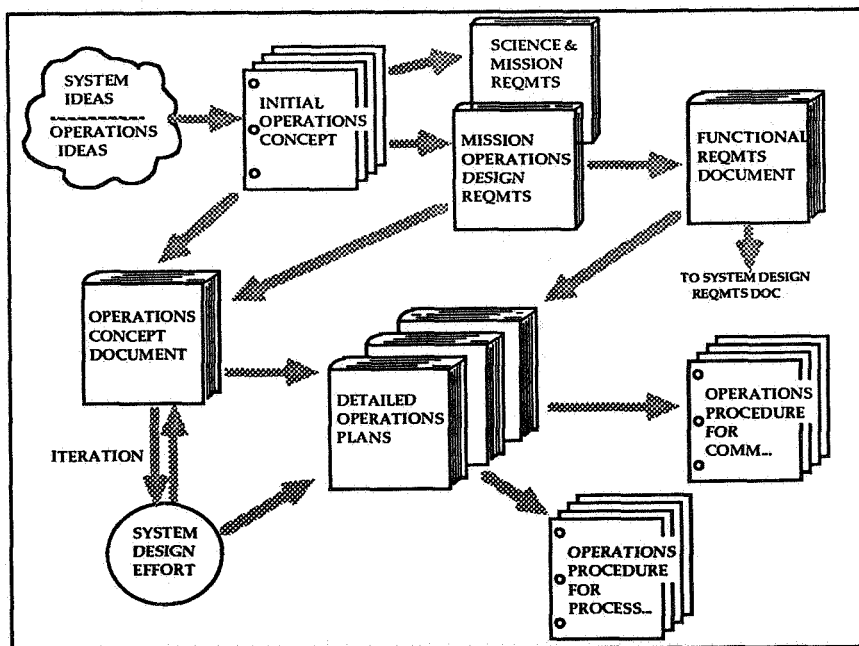


Figure 2 - Operations Planning And Development Flow

1. *Obtain early agreement on basic operational approaches.* It is more important that all the players agree on an approach, even if operationally less-than-optimal, than to have a perfect approach.

2. *Keep the document concise yet comprehensive.* It should be a summary of intended operations. It must cover all areas of operations necessary to accomplish the mission, but restricting details in each area to that essential to conveying the message.

3. *Keep it updated.* Revisions or additions, scheduled after major steps in the planning, design and

1. Introduction	Scope and Objectives of Document Organizational Roles and Responsibilities
2. Summary	Flight Vehicle and Ground System Configuration
3. Objectives, Constraints and Approaches	Mission Objectives and Constraints Experiment Operations Objectives Approaches to Operations Activities
4. Operations Requirements and Functions	Summary of Operations Requirements Functional Definition and Operations Flows Definitions of Operational Modes
5. Operational Activities and Scenarios	Activity Table and Timelines Normal Operational Scenario
6. Contingency Operations	
7. Operational Interfaces	Interfaces External to System Interfaces Internal to System Human-Computer Interfaces
8. Operational Hardware Features	User Equipment & Operations Console Features Communications Nets and Data Links Special Facility Requirements
9. Personnel Component	Operational Positions and Responsibilities Organization and Support Coverage Training Concept
10. Reliability, Maintainability and Availability	System Reliability and Availability Maintenance and Logistics Concepts

Figure 3 - Concept Document Content

implementation processes, are essential. To be useful, it must be current. The design team will ignore an obviously out-of-date Concept. Up-dates released at each major review in Figure 1 would be preferred.

4. *Let it evolve with the design.* Although the concept will levy operations requirements on the design, the concept should be allowed to change (configuration controlled) if the design effort indicates more efficient or less costly ways of implementing the requirements. In some cases, the requirements themselves may also evolve with the program.

5. *Keep its focus user-oriented.* The focus must be on the eventual operation of the MOS to achieve desired mission objectives, and on the users that must operate it.

4. REQUIREMENTS AND DOCUMENTATION.

In most programs, the definition of the initial operations concept usually occurs in parallel with the formulation of top-level mission and science requirements that will later govern the design. The Operations Concept, while not a requirements document per se, must be translated into requirements specifications that will implement the desired concept. Requirements specifications are essential to establishing a

common understanding between customer and contractor as to what the task must include and what the job must accomplish. Agreement on the requirements before proceeding to design is critical to efficient, cost effective, and on-time development of a system that will meet the customer's needs. In addition, requirements documents provide a source of test objectives for the program test phase as well as providing a valuable archival reference during the remainder of the program. For these reasons, the Operations Concept developer must understand the various types of requirements that are levied both on spacecraft design and on a MOS.

Mission requirements are high level statements of the goals and objectives of the mission itself, or in other words, what it is that the mission is required to achieve. Mission requirements may reflect such issues as the type of orbit necessary, the mission duration to achieve the objectives, number of spacecraft contacts per day, and/or spacecraft pointing accuracy.

Parallel to these are the *science requirements*, which define the goals and objectives of the science experiments, in many cases further delineating the mission requirements. Examples of these are specifications of the target characteristics to sense, the resolution of the data to be obtained and the frequency range of an instrument.

Operations requirements are MOS design requirements relating specifically to the ways of achieving the operational mission. In general, they define the scope of the ground activities within the MOS. They specify issues such as the number and type of ground antennas for spacecraft contact, the necessity for around-the-clock monitoring of vehicle activities, the types of computational activities that must occur, how experimenters will gain access to their data, and whether or not command sequences are validated with a simulator. At this level, the ground system can be treated as a series of black boxes, where the operations requirements define the functionality of the box (what it needs to do) and its interfaces with other boxes, but do not delve into the details of how the correct product is achieved. However, to design the black boxes, each operations requirement must be decomposed into one or more functional requirements.

Functional requirements are those that govern the design of the system components (to satisfy science, mission, or operations requirements) by identifying the specifications for each function the component must accomplish. The primary emphasis of functional requirements is on *how* the system is implemented rather than what the component does or the content of its operational product.

Lastly, *performance requirements* are those which specify when functions or activities must be completed, how fast and their duration. They delineate the order in which tasks must be performed, the amount of time allowed for an activity to be accomplished, other activity milestones that must precede or follow given activities, and resource constraints imposed, especially when computer operations are involved.

Figure 4 is a document tree for a typical spacecraft mission operations system. Not all of these documents are required for every project, nor are all the documents shown that a given project may need. Those documents on the far right are those developed during the design and build of the spacecraft but which are essential to retain and to *maintain* during operations. They do not fit specifically into a MOS document tree, but instead supplement it. To the left are those documents that are developed with the operations phase of the mission in mind, although they may be written very early in the life of the project. Along with the initial concept of operations, both the mission requirements and the science requirements are derived. The MOS Design Requirements document specifies how the mission and science requirements are converted to operations requirements to achieve implementation of the spacecraft operations activity. These, in turn, are decomposed into the functional requirements which are recorded in FRDs for both team activities and for the ground data system. For the latter, functional requirements documents are followed by hardware and software requirements and design documents and either

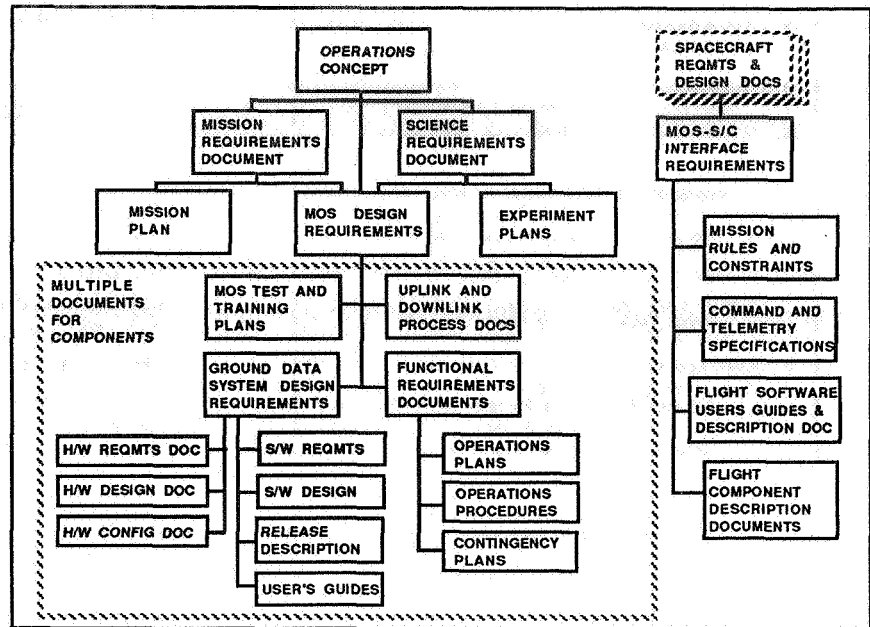


Figure 4 - Typical Document Tree

configuration documents or users guides. For the teams, the FRDs are followed by operations plans and procedures.

5. SUMMARY.

This paper has defined an Operations Concept as a part of a structured process for design and development of a mission operations system (MOS). It has discussed the program phases, required reviews and documentation necessary to achieve a complete, efficient and cost-effective MOS. It emphasized the importance of the Operations Concept, written early and maintained, and stressed the need for complete definition and agreement on requirements before the design has proceeded too far to easily modify. In short, a well-written, complete and maintained Operations Concept document will contribute significantly to a well-designed, efficient and cost-effective MOS.

6. REFERENCES.

1. Wall, S. D. and K. W. Ledbetter, *Design of Mission Operations Systems for Scientific Remote Sensing*, Taylor and Francis, London and Philadelphia, 1991. The majority of the material in this paper is taken from Chapter 2 of this textbook.
2. U.S. Air Force Regulation 57-1, Attachment 5, *Operations Concepts*, 1977.

MISSION ENGINEERING

N94-23881

Paul Ondrus
NASA's Goddard Space Flight Center
Greenbelt, Maryland, U.S. A

Michael Fatig
Bendix Field Engineering Corporation
Seabrook, Maryland, U.S.A.

ABSTRACT

Goddard Space Flight Center's projects are facing new challenges with respect to the cost effective development and operation of space-flight missions. Challenges, such as cost limits, compression of schedules, rapidly changing technology, and increasing mission complexity are making the mission development process more dynamic. This paper proposes a concept of "Mission Engineering" as a means of addressing these challenges. It is an end-to-end, multi-mission development methodology that seeks to integrate the development processes between the space, ground, science, and operations segments of a mission. It thereby promotes more mission-oriented system solutions, within and across missions.

Key Words: Aerospace, Unmanned Scientific Satellites, Mission Development

1. MISSION TRENDS

As technology rapidly changes in both onboard

and ground systems, and as science moves towards more dynamic process studies, our missions, their operations, and the support systems are constantly evolving. Understanding these trends and managing change throughout the end-to-end system in a controlled and collaborative manner is essential to cost effective obtainment of mission objectives. This creates a challenging environment for GSFC projects.

Five mission trends that are contributing to GSFC project challenges are the increase in real-time science operations (moving from survey to dynamic process studies), fundamental changes in ground/space asset relationships, the movement towards distributed processing environments, the demanding GSFC mission model, and trends towards small spacecraft series.

First, the present mission model shows a trend from survey-type missions to missions and sets of missions that involve more precise, dynamic study processes. This is resulting in higher

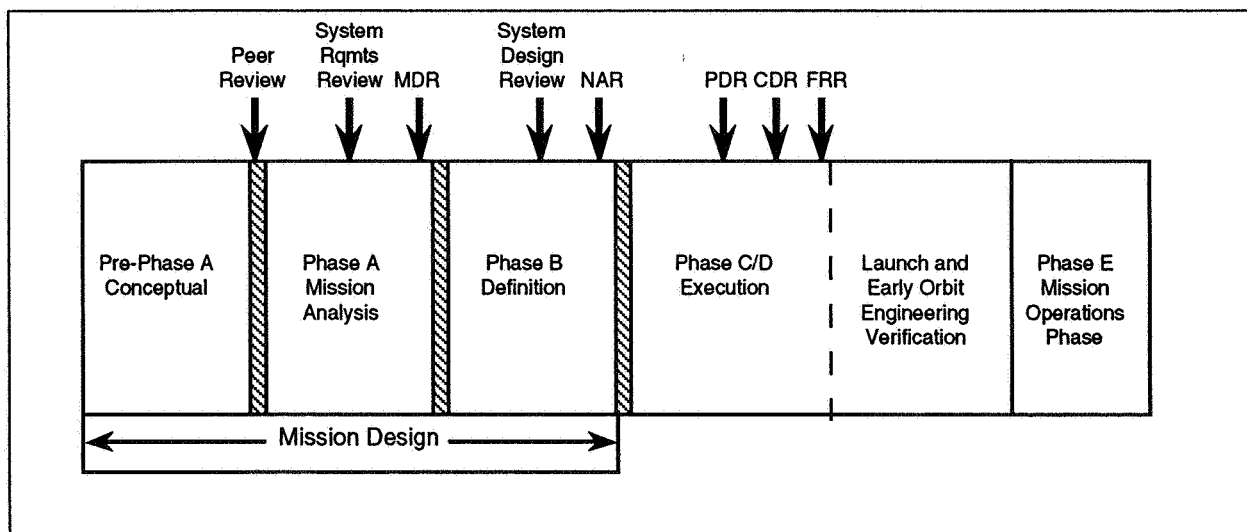


Figure 1: GSFC's Mission Life Cycle

burst data rates, less advanced planning, more targets of opportunity, correlative science operations, and coordinated campaigns between spacecraft missions. This creates increased demands upon the space, ground, and operations segments of a mission and implies a closer coupling of spacecraft and science operations. Science operations are therefore becoming more of an integral part of the planning and realtime systems.

Secondly, the availability of more powerful flight processors is allowing the movement of more functions on-board and allowing more flexibility and autonomy in mission operations. This is evolving the relationship between the ground and space segments, moving away from a ground master, spacecraft slave relationship to a more peer-to-peer level of operations. This requires more coupled engineering of space and ground segments, and changes the nature of normal and fault condition operations.

A third trend is the growing distribution of ground and science systems that support a mission. Today's technology is emphasizing moving data rather than people using data driven processes. This trend requires more up-front operations and systems planning to ensure that the mission objectives can be met while handling the contingencies that occur in the course of a mission. This trend places more emphasis on the early definition of information exchange, interface design, and element interaction.

The result of these three trends is a tighter

operational coupling of the space, ground, science, and operations elements during operations, and therefore requires segment coordination within an end-to-end mission system earlier in the mission development process. Failure to do so can result in costly modifications and redesign, and use of systems in ways not intended (causing system stress and error potential).

Finally, the GSFC mission model and the trend towards smaller spacecrafts in a collaborative series (versus the previous single, large spacecraft approach) has resulted in many simultaneous mission processes and common operations and facilities for spacecraft series. It is now necessary to create a conscious, structured process for working across missions to promote reuse and sharing, thereby achieving additional economies.

In an effort to met these and other challenges the concept of a Mission Engineering process is provided. This concept provides the necessary interactions between the elements to ensure that mission trades are conducted (building the best "mission" solution), exchanges of information are understood, planned, and performed, and the systems are designed for their intended use.

2. WHAT IS MISSION ENGINEERING?

Mission Engineering is fundamentally a process improvement concept that addresses the interactions between the main mission elements

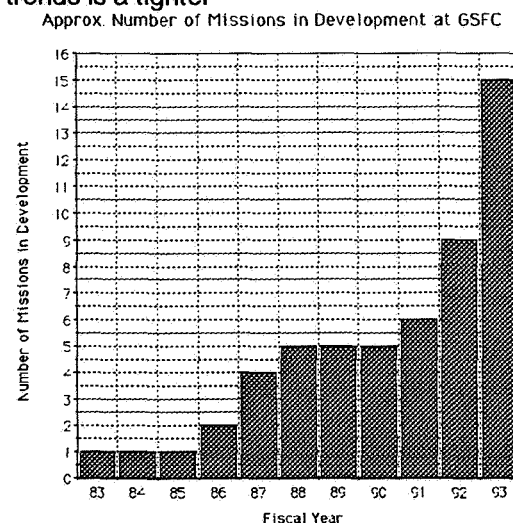


Figure 2: GSFC Mission Development Model

of space, ground, operations, and science. It is a structured, conscious effort at integrating the traditional space and ground segment development activities, applying the operations engineering principles, and embracing the inclusion of the now realtime science element in a development and operations process over the complete mission life-cycle. It also provides a structured communication mechanism for addressing mission trades early in the mission development process (particularly during the critical Phase A and Phase B study efforts where fundamental strategies and budgets are established), a means for developing and applying new technology and standards in a controlled manner, and a means for applying past lessons learned for continuous improvement. Its emphasis is on understanding the overall operational and science vision and in sharing that vision across the main mission elements.

Given the cost constraints of today's environment, the Mission Engineering process seeks to make the best use of standards, policy, and

lessons learned from previous missions in order to identify the key mission drivers and to address them in a timely manner. It focuses upon applying the best skills and experience to the right problems in order to quickly resolve operational issues, address life-cycle costs, and to ensure that the correct system is built the first time. The key is to understand and discuss the essence of the problem and not create a bureaucratic process that hides key issues.

A key objective of this concept is to enhance interelement communications and to ensure that operational impacts of system design decisions are understood. Communications is a key element because the process emphasizes the need for concurrent mission definition activities in order to compress the development schedule for smaller missions. This works to anticipate the mission needs with respect to reusability of ground and science components and proven concepts. The key is not to recreate infrastructure, but to focus project resources on the science that is to be obtained.

2.1 Dimensions of Mission Engineering

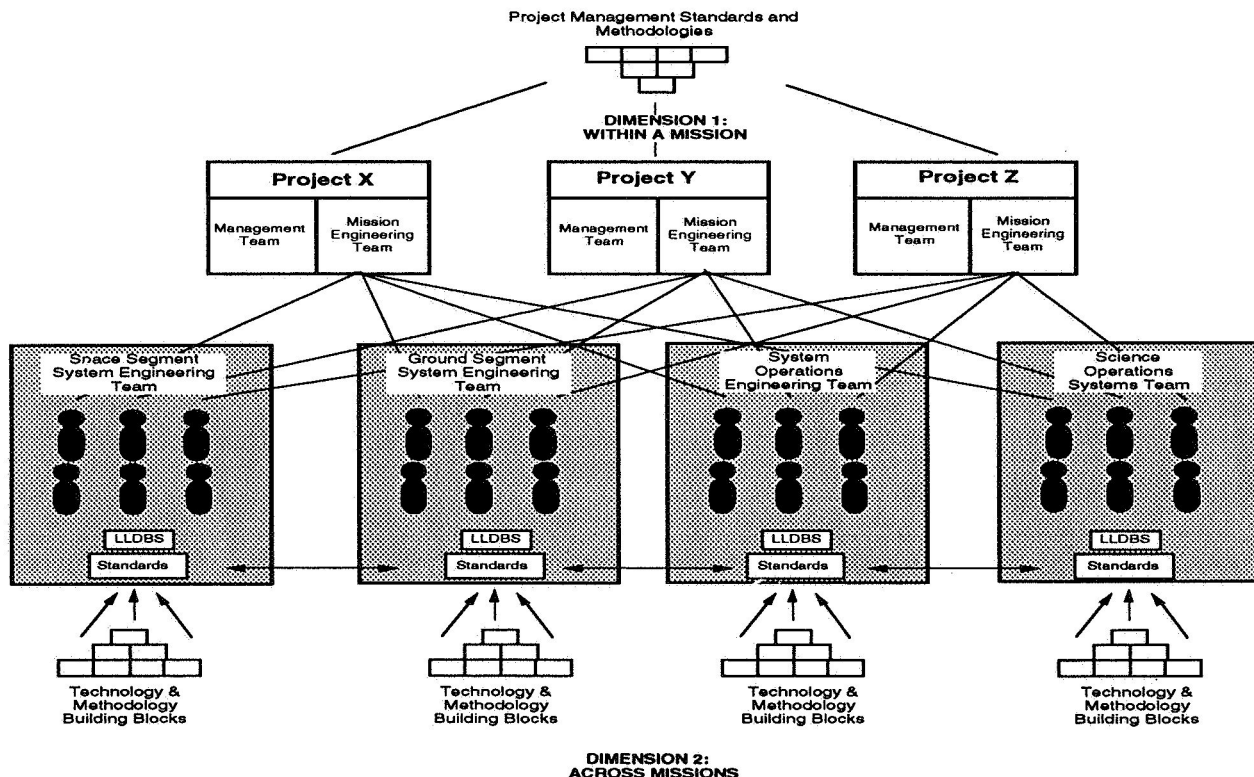


Figure 3: The Mission Engineering Architecture

The Mission Engineering concept has two dimensions: 1) within a mission and 2) across missions. These two dimensions are active simultaneously and interact with each other.

2.2 Within A Mission

Within a mission, the Mission Engineering process emphasizes addressing mission and science operations in a structured manner early in the development process. It requires a mission engineering team (space segment, operations, and ground system engineer plus science representation). The objective is to trade ground, operations, and science alternatives in a concurrent manner with spacecraft design efforts with the emphasis placed on mission management team defined and weighted decision criteria.

The process at this point is driven by a process guideline to ensure completeness in order to address all mission aspects. It seeks to develop and maintain a complete operations concept or vision of the mission and to communicate this to all elements. In many of today's missions this vision is baselined once and left in Phase A. The Mission Engineering process is based upon the concept of applying operational experience and science expertise early in the mission

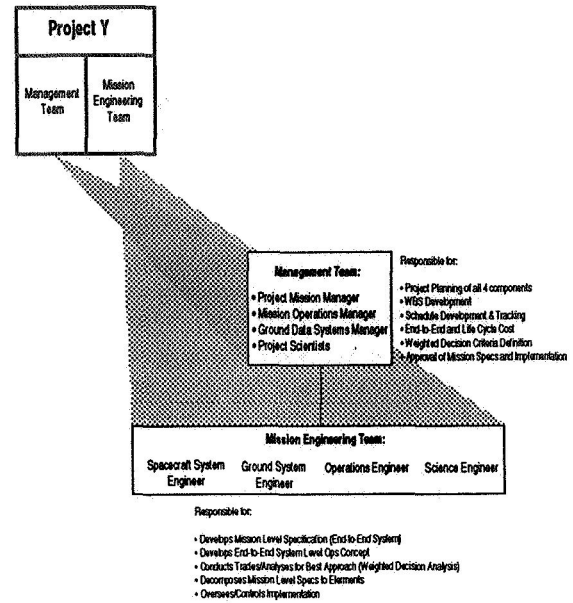


Figure 4: Dimension 1 Organization

development phase in order to draw from lessons learned on previous missions. It also provides a means for injecting through these individuals existing capabilities in the ground and science infrastructures.

History has proven that it is more cost effective to address these issues and resolve them early in a mission rather than later. Unfortunately, in

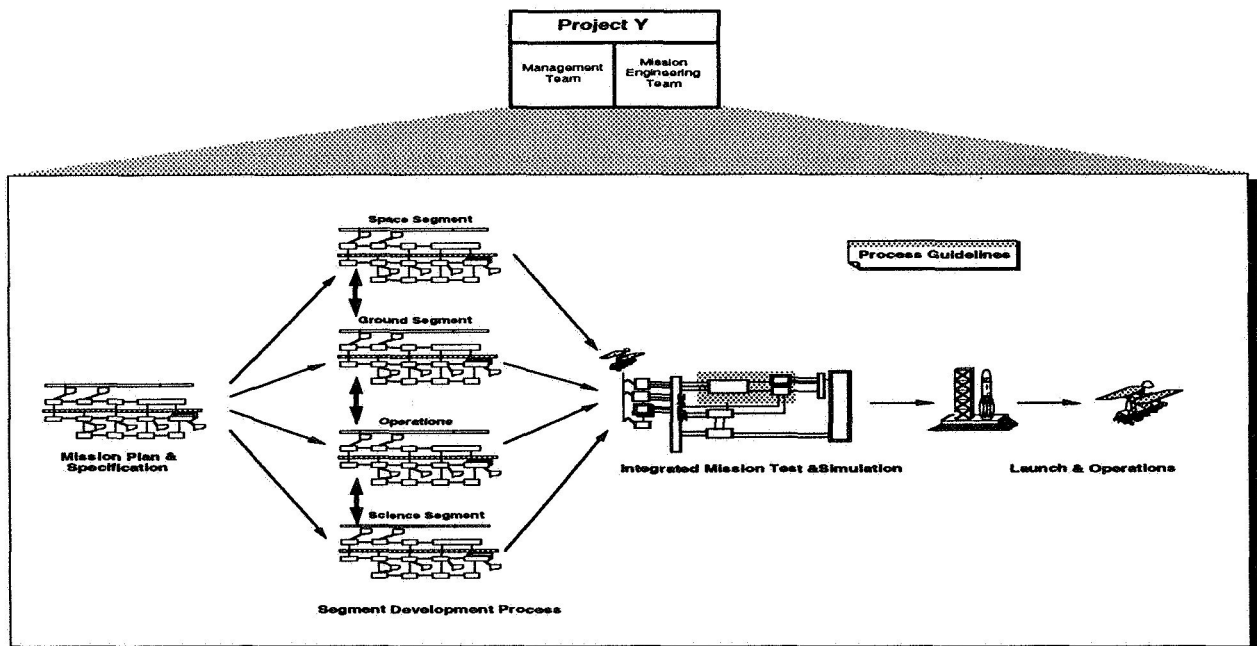


Figure 5: Dimension 1 Process

many missions they are addressed late and cause increased costs and operational readiness delays. Mission Engineering encourages that the right questions be asked up-front and that contingency scenarios be addressed. Without applying the proper skills and experience the same mistakes are made from mission to mission. These problems are subtle enough so that they are only discovered during the end-

major launches occurred. Therefore we have been lulled into a process which did not promote across mission engineering, nor necessarily need it. However, today the mission load is quite different, with many major missions scheduled for launch from 1994 to 1998 and beyond. We must now find ways to improve our processes for looking across missions which are in parallel development stages and search

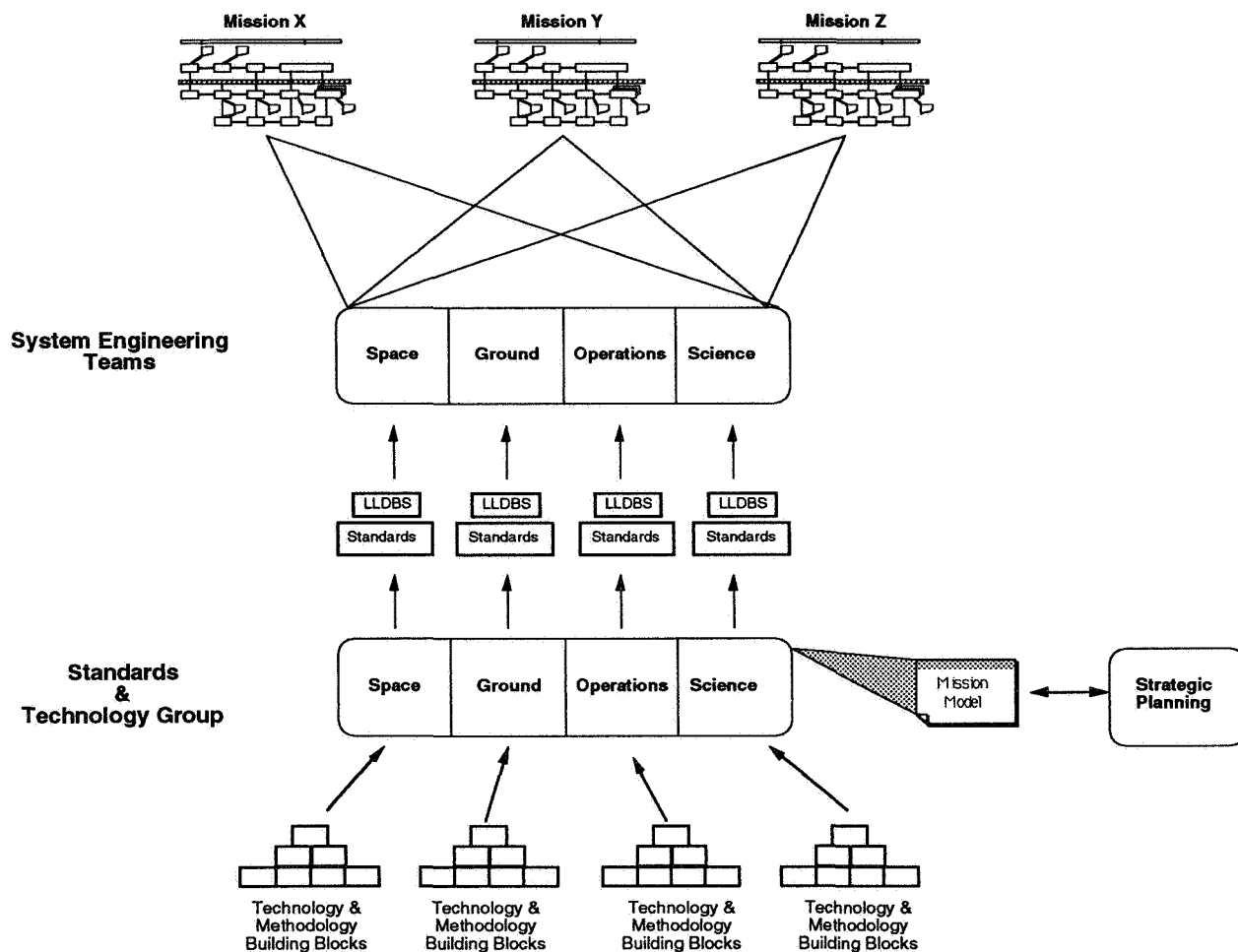


Figure 6: Dimension 2 Organization and Process

to-end testing phase rather than at reviews.

2.3 Across Missions

GSFC has been successfully implementing individual missions for over 30 years. Within the past ten years the mission load has been relatively low, with major free-flyer launches at a rate of one or two per year. During the period from 1986 (Challenger disaster) to 1989 no

for additional economies and common solutions. The Mission Engineering process therefore provides a means for integrating development and operations across missions, complimenting the individual "within a mission" dimension. The intent is to promote the sharing of technology, applications, and operations across missions so that each separate project does not reinvent the wheel nor implement a different solution for a like function. Other less challenging but eco-

nomical benefits such as mass buys can also be obtained.

The products of this element (maintained by a multi-segment standards and technology group) are mandated standards, reusable ground and science software, multi-mission technology, and requirements for the support infrastructure. Also provided is a central repository for lessons learned, a forum for operational issue resolution, and the development of operational personnel. An overall mission model is maintained and is used to articulate noted trends back to the various groups, as well as to strategic planning elements. Multi-mission engineering is then imparted to active mission development processes by multi-segment systems engineering organizations using common standards and technology. It is executed by a senior system engineering team representing the space, ground, science, and operations elements.

2.4 Why Mission Engineering?

The lack of a conscious, structured multi-mission process for projects leads to a significant increase in the overall mission budget. Lessons learned and technology gained on one mission will not be systematically shared with other missions if there is no mechanism for sharing such information. Solutions to common problems will be applied differently, limiting reuse and the economies that come with it. Individual missions will provide a suboptimization of agency resources. There needs to be a proactive advocate that can encourage projects to reuse systems and operations across missions. Due to the rising MO&DA budget and the agency attention at reducing operations costs, a focus needs to be provided that understands the implications of operational concepts, spacecraft design, and infrastructure changes. Without such a focus the operations budgets will continue to grow and in times of constrained budgets new and exciting science missions will be prevented from taking place. Mission Engineering provides a forum for preventing future operations increases.

Presently, there is no element responsible for understanding the strategic implications of mission model changes. A good example is the overwhelming trend in flight projects toward small, ground station-only spacecrafts, yet the

operations support capabilities continue to focus critical mass on the TDRSS system with no major plans for ground station enhancements. A similar divergence exists in the trend towards higher data rates and more realtime data requirements (in support of dynamic process science studies), yet no major changes in the data transport capabilities seem imminent. The systems and commercial enterprises that comprise NASA's communication and data acquisition infrastructure take years to change. The analysis of the mission model and science needs has no focus to presently map future needs into these systems. Technology is changing faster than the infrastructure can be cost effectively changed. The Mission Engineering forum provides a mechanism for attempting to anticipate these types of changes.

This process and its corresponding personnel provides a means of improved communications between organizational elements at GSFC and provides project managers a forum to have operational system's issues analyzed. It will improve communications which in turn will improve the timeliness and quality of the products provided. It also has side benefits of promoting team development and increasing the overall knowledge of each member of the team to the sensitivities, cost drivers, and needs of the other. It will enhance a "one team" spirit among the center's directorates. Finally, given a documented structured process for mission engineering, and the ability to capture and apply lessons learned, the concepts of continuous improvement can now be better applied.

3. SUMMARY

Mission Engineering is a process improvement concept that is beginning to be implemented at GSFC. A working group will soon be formed to develop the joint process. Steps are presently underway to address the skills needed to assemble an effective dimension 1 (within a mission) team, and a joint interelement working group is being explored to formally address the multi-mission dimension. We feel this concept will address the many challenges facing us including the need for "more for less".

MANNED SPACE-LABORATORIES CONTROL CENTER (MSCC) OPERATIONS CONCEPT

Dr. Joachim Kehr
Deutsche Forschungsanstalt für Luft- und Raumfahrt (DLR)
D-8031 Oberpfaffenhofen, Germany

N 94 - 23882

ABSTRACT

The paper will recall the initiation of the (German-) nationally funded control center for manned spaceflight operations triggered by the invitation of President Reagan to ESA, Japan and Canada in 1984 to join the International Space Station Freedom Program.

The requirements for a Manned Space-Laboratories Control Center (MSCC) as defined at the beginning of the planning and construction process in 1987 and the resulting modifications during the various programmatic scenario changes on NASA and ESA side between 1987 and now will be presented.

The validity of the original requirements with respect to the current scenario, which asks for a logical evolution from the execution of the D-2 mission in January 1993 via the European Columbus Precursor flights (in particular the E-1 mission) towards Columbus Attached Laboratory (APM)-operations by the end of this century will be discussed.

The resulting tasks of the MSCC for the various missions, the current configuration and the ensuing operations concept leading from a more centralized concept for D-2 towards a decentralized payload operations concept for the APM and the implications with respect to European and International interfaces will be presented.

The planned Columbus MSCC facility architecture and its expected modifications introduced by the ESA Ministerial Conference in Munich (Nov. 1991) and follow-on discussions will be briefly addressed.

The last chapter will outline the planned services to be provided by the MSCC to the decentralized User (experimenter) community. Issues like decentralized mission planning on executional level, command validation, data flow coordination, archiving services and telescience capabilities will be highlighted from a MSCC point of view.

Key Words: Columbus Operations, Manned Space Laboratory Control Center, User Operations Support

1. MSCC FACILITY DESCRIPTION

The Columbus program originally consisted of three elements ("Den-Haag Scenario"):

- an Attached Pressurized Module (APM) to the Space Station Freedom to be launched in 1996 by the Shuttle
- an European Free Flying Laboratory (MTFF) in a similar orbit as the Space Station to be launched with an ARIANE-5 booster in 1998 and serviced by HERMES from 1999 onwards.
- a serviceable observation platform (PPF) in polar orbit to be launched by ARIANE-5 in 1997

During the ESA Ministerial Conference in November 1987 at Den-Haag, it was decided to develop a decentralized European ground infrastructure, Germany being entrusted with the built-up of the control center for the two manned Columbus elements.

Columbus Experimentation and MTFF Operations would be accommodated in the nationally funded and constructed Manned Space Laboratories Control Center (MSCC) with its arrangement of operations rooms and office space as required for the operations and support functions of the original Columbus scenario:

- Payload operations coordination for all the European experiments to be carried out on-board the Space Station Manned Base (SSMB), as well as safety and health insurance of all European-provided payload facilities/instruments located in the APM or elsewhere in the SSMB,
- Mission control functions for the Columbus Free Flyer Laboratory and Resource Module subsystems, payload operations monitoring, control and/or coordination functions for all

on-board payload equipment and experiments,

- Ground operations functions for the implementation, preparation, operations and maintenance of MSCC subsystems and external space- and ground-network planning and execution coordination,
- Training (T) functions for MSCC ground personnel,
- Qualification & validation (O & V) functions of the MSCC for supporting the above functions, and
- Related facility, administrative, financial and personnel managerial functions.

The MSCC, as completed in 1989 consists basically of two buildings, the Mission Control Building and the Operations Mission Sequence Simulator (OMSS) High Bay area, which are connected via a Foyer. A third building, the In-Orbit Operations Simulation Facility (IOSF), is also connected to the Foyer, but is dedicated to manned spaceflight technology developments including the European Proximity Operations Simulator (EPOS) and Servicing Test Facility (STF). The IOSF is independently managed and operated and is not addressed further.

The Mission Control Building accommodates the flight and ground operations control / coordination functions and facilities.

The OMSS High Bay area would accommodate the majority of the TQ & V tools and simulation capabilities for the two flight elements. The simulation tools are MSCC "in-house" tools which will be used to support independently from external sources both test and validation activities for the MSCC facilities, as well as training of MSCC Operations personnel.

2. D-2 / E-1 OPERATIONS

The MSCC is presently configured to execute the second German Spacelab mission (D-2) in spring of 93. For D-2 the MSCC will act as Payload Operations Control Center (POCC) which is different from its Columbus role as European Payload Operations

Coordination Center (E-POCC).

The difference being that for D-2 all Experimenters are assembled at the MSCC for the duration of the 9-day mission to perform all experimentation activities while in the Columbus era the experiment operations will be conducted from the Experimenters' home bases.

Besides the experimenters' participating remotely in Columbus operations, the significant difference to D-2 is, that Europe will only utilize part of the total Space Station capabilities, i. e. the European utilization planning has to be integrated with that of the other partners. This overall integration is performed by the "Payload Operations and Integration Center" (POIC) at Marshall Space Flight Center (MSFC).

While D-2 planning could optimize the resources with "custom tailored" timelines, Columbus will have to utilize "resource envelopes" in order to allow decentralized utilization planning in Europe and all the other partner countries. An evolutionary process from the centralized D-2 concept towards the decentralized Columbus operations scenario has to take place. The interconnecting link will be an Europeanized Spacelab flight with international participation, called Columbus Precursor Mission E-1, in 1997. Since no experience exists on planning and operations schemes like that envisaged for Columbus, the goal of the E-1 mission will be to gain knowledge and exercise operational management arrangements and procedures for:

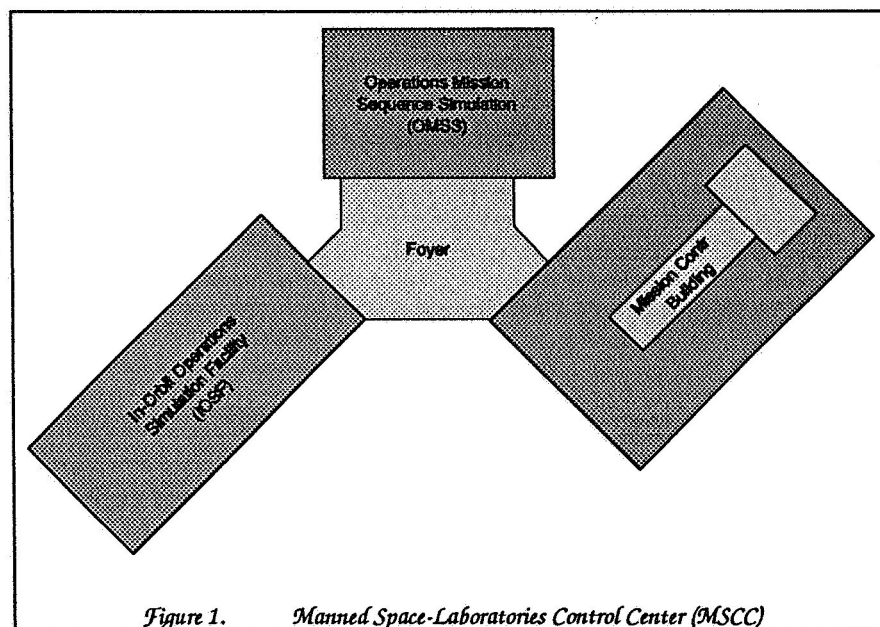


Figure 1. Manned Space-Laboratories Control Center (MSCC)

- payload operations planning on executional level in Europe by employing resource margins
- decentralized payload operations with enhanced telescience capabilities by minimizing interactive time delays
- direct high-speed data services to the user home bases with intermediate data storage capability at the MSCC.

This will change the focus of the MSCC from a "CONTROL" Center to a European resource- and command-"COORDINATION" Center.

So far no major impact of this change in focus on the functionality or the physical structure of the MSCC as described in Chapter 1 have been identified.

3. COLUMBUS OPERATIONS

Due to various de-scoping- and rescheduling exercises of the SSMB by NASA and due to the dramatic political developments in Europe during the recent years, in particular the German unification, the disintegration of the USSR and the ensuing opening of Europe towards the east, the European (ESA-) Long-term plan had to be readdressed. In the aftermath of the ESA Council Meeting on Ministerial Level in Munich 1991, also the Columbus program was considerably descoped:

3.1 Space Segment Baseline

The current MSCC Operations Concept takes the "Munich Conference" and follow-up decisions into account which are with respect to the MSCC:

- No Columbus Free Flyer has to be considered any longer
- No Hermes back-up functions have to be provided by the MSCC or vice versa
- APM launch in 1999

The following main space segment baseline items are currently considered in the MSCC design:

European Payload on board the SSMB

- | | |
|---------------------|------------------------------------|
| • Launch | Nov. 99 (NSTS) |
| • APM integrated | 4 t P/L mass |
| • Utiliz. Flights | 3 UF (t.b.c.) |
| • Nr. of increments | 4....5 / year |
| • APM length | 4 x 8 double rack |
| • External | separate launch |
| • Viewing platform | 1.3 t P/L mass
retrieve by NSTS |

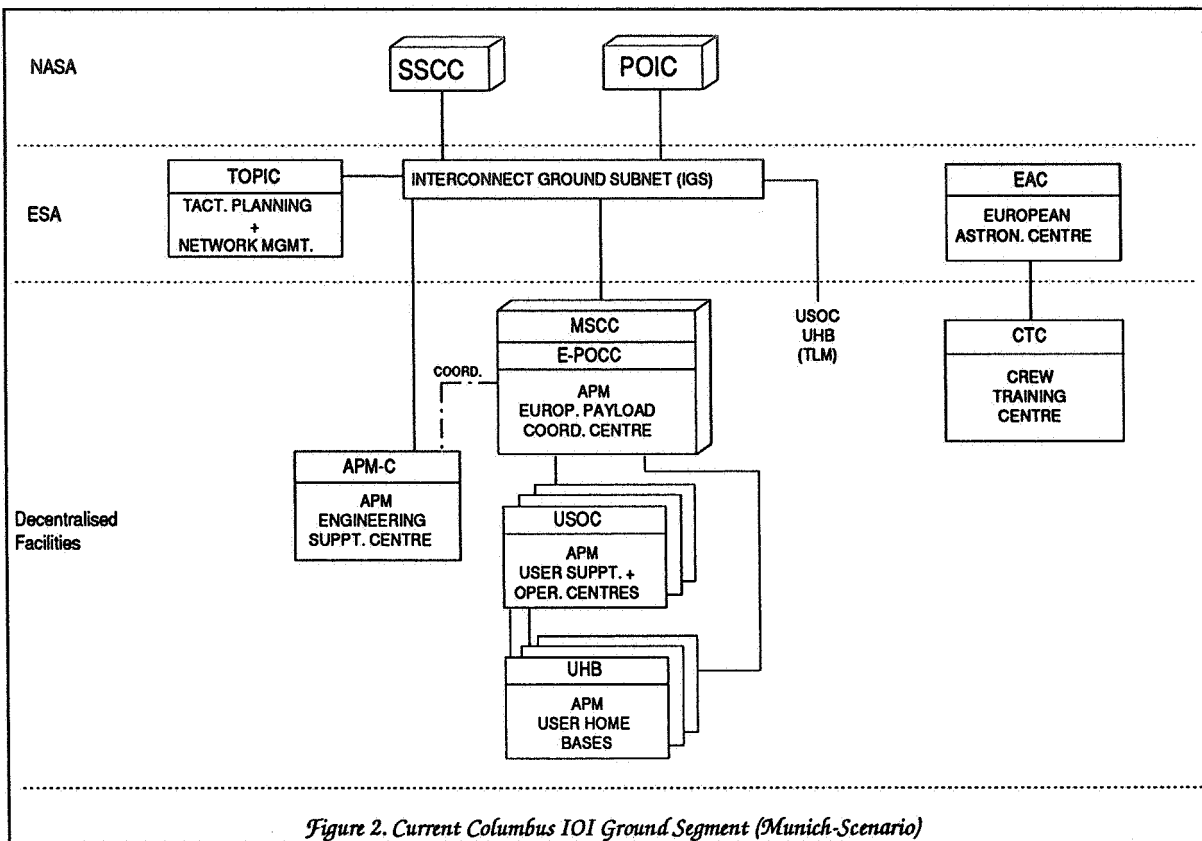
3.2 Current European In-Orbit Infrastructure Ground Segment

The original decentralized European In-Orbit-Infrastructure (IOI) Ground Segment (GS) set-up included a variety of existing and planned ground facilities in the ESA Member States under the overall control of a central ESA tactical planning and monitoring authority:

- the Central Mission Control Centre (CMCC) at ESOC in Darmstadt for APM and MTFF tactical planning tasks, coordination and control of mission activities involving more than one European element, central navigation support and management of communication resources management
- the MSCC for APM Payload and MTFF Payload / System Operations
- the HERMES Flight Control Centre in Toulouse
- the Polar Platform Control Center
- Engineering Support Centres for payload integration, engineering support and logistics services for the APM, MTFF and Hermes, located in Italy, Germany and France
- an European Astronaut Centre with associated Astronaut Training Facilities in various member states
- DRS Operations Control Centre in Italy
- AR-5 launch and landing facilities

In addition a range of User Support and Operations Centres (USOC's) and User Home Bases (UHB's) for APM and MTFF payload operations in many ESA member states were planned. With the described descope measures the above infrastructure had to be simplified and is depicted in Fig. 2.

The MSCC role for the "Munich"-Scenario is described in the remainder of this paper.



3.3 MSCC Role

The MSCC will be responsible for European experimentation activities coordination on board the SSMB i.e. for all European Payloads and Experimentation onboard the Columbus Attached Laboratory or elsewhere on the Space Station the MSCC Payload Operations Coordination function will be:

- Act as the Single Point of Contact between the European User Community and the POIC
- Coordinate and Integrate formalized User Requirements
- Provide Planning Inputs to the POIC in form of an European Increment Operation Plan (IOP) for the overall IOP Generation, and as Activity Models for the Short Term Plan (STP) Generation
- Provide coordinated European Replanning Inputs to the POIC for the IOP and STP
- Ensure the Resource Availability for European Experimentation during the planning phase
- Enable/disable command capability for European users, and validate commands,

according to plans or on requests, in close cooperation with the POIC

- Monitor and control Experiment Resource Envelopes during operations
- Ensure Health and Safety of European Payloads and take appropriate steps of coordination
- Coordinate and Ensure End-To-End Data Handling for all European Experimentation
 - Coordinate Voice Communication between Crew and authorized User
 - Coordinate RT and Offline Data Dissemination
 - Provide Data Storage for Low Data, Voice and Video and High Rate Data in Backup Case
 - Distribute recorded data offline
- Provide Post Increment Reports
- Accommodate users which are not supported by an USOC

3.4 MSCC Operations Team Structure

The shown MSCC Team Structure is derived from the role as described in the previous chapter for the APM Payload Operations Function.

The structure is based on the agreements as defined in ESA/C(87)84 "Ground Facilities for Operations" and the Den Haag Resolution ESA7C/LXXXIX/Ref.1 "Resolution on the ground segment associated with the European in-orbit infrastructure operations", i.e. for each increment:

- the "delegated" operational tasks on execution level will be executed by a DLR Flight Director (approved by ESA)

- tactical level representation will be performed by an ESA appointed Representative possibly with staff on site.

Overall MSCC executional management will be performed by the DLR MSCC-Project Manager. The staffing profile for the operations teams will be optimized to allow the execution of the current increment and preparation of the next ones simultaneously.

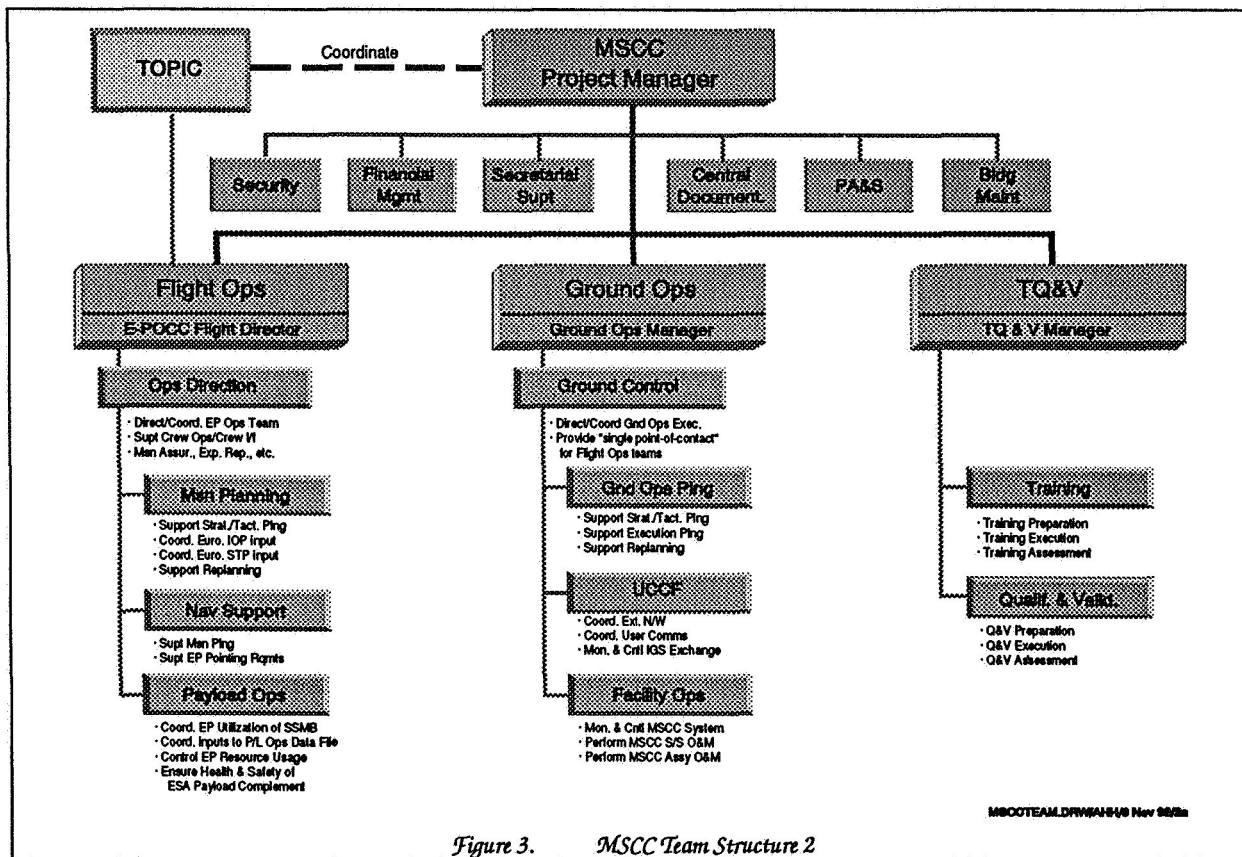


Figure 3. MSCC Team Structure 2

3.5 MSCC External Interfaces

An overview of the MSCC external interfaces is provided in Figure 4.

3.5.1 US Interfaces

A preliminary definition of the interfaces to International partner facilities has been developed. These interfaces are the subject of the Functional Control Documents (FCDs), which document the agreements between the US and the international partners. Although these documents are still in

the process of being developed and negotiated, general description of the US interfaces is provided.

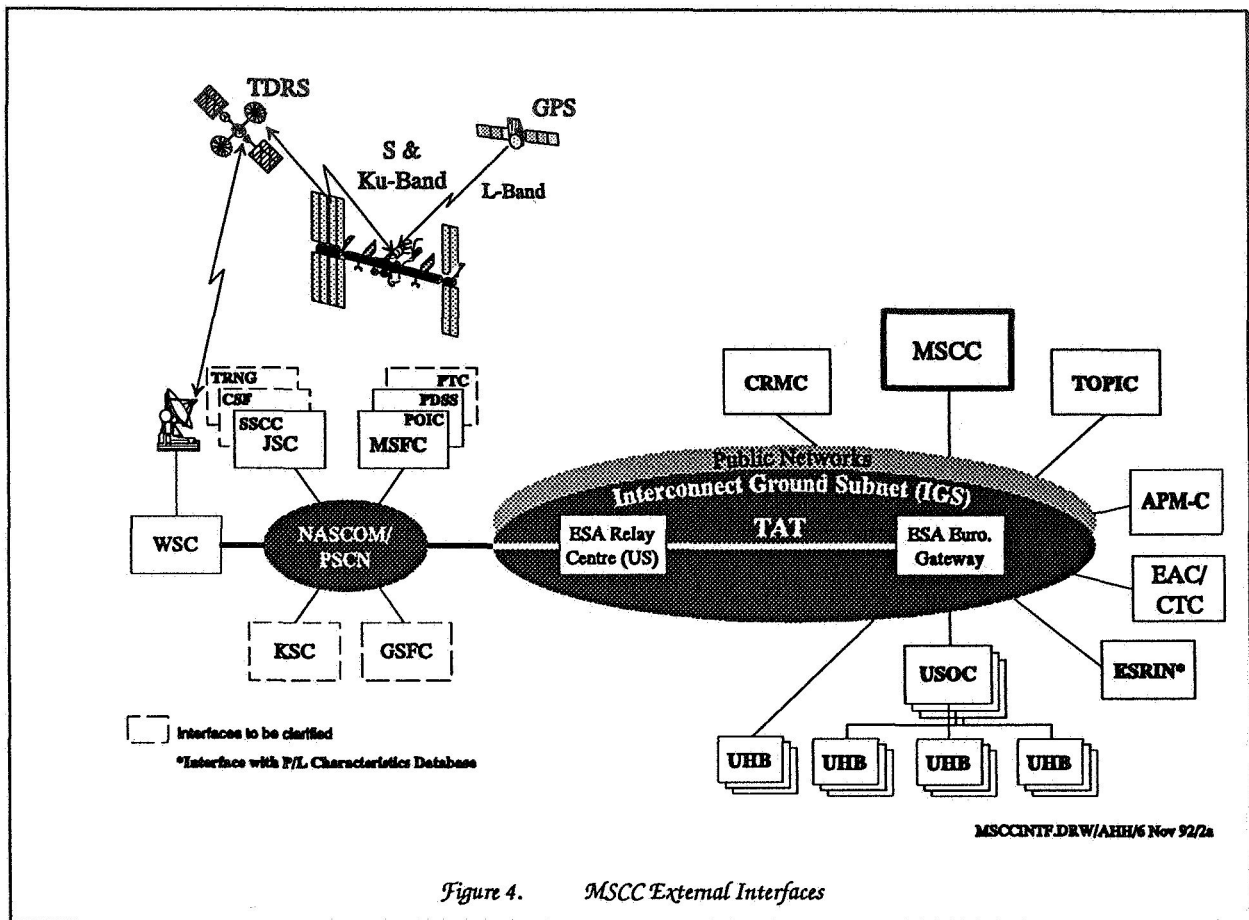
POIC Interfaces

The major POIC functions with respect to European Payload Operations are:

- coordination of NASCOM data services
- command validation
- generation of hazardous P/L commands
- integration of mission planning / replanning inputs (IOP, STP)

• coordination of Air-to-ground voice com-

allow the MSCC to access information and



munications

- SSMB resource coordination for all payloads
- assurance of safety and health for all payloads
- overall responsibility for servicing and trouble shooting activities (with MSCC support)

SSCC Interfaces

Although the POIC is the point-of-contact for coordination with SSMB system operations, the MSCC will establish interfaces to the SSCC for the delivery of selected planning and preparation data, and for the reception of broadcast video and systems data. The coordination of these data transfers will be performed over coordination voice loops with POIC, which extend to selected areas in the SSCC. In addition, access to the NASA electronic information System TMIS will

services support at Johnson Space Center.

3.5.2 European Interfaces

The MSCC will establish interfaces to user facilities and other IOI GS facilities. The majority of these interfaces will be provided by a ESA Interconnection Ground Subnetwork (IGS), however non-operational interfaces and non-real time interfaces to users will be established through public communications networks (local PTTs). Unlike the NASA interfaces, which require negotiation and agreement between ESA and NASA and are documented in the FCD's, MSCC interfaces to facilities in Europe will be developed during the design phase, agreed upon by the facilities involved, and documented in separate Interface Requirement Documents (IRD's). Such interfaces have to be established to Tactical Operations and Integration (TOPIC) Function, APM Engineering Support Center (APM-C) and to the European Astronaut Train-

ing Center (EAC) with associated Crew Training Center (CTC).

4. MSCC SERVICES FOR USERS

The decentralized Columbus payload operations concept provides opportunities for the users to operate and control their experiments from their home institutes / home countries (USOC's and UHB's) with less stringent conditions than during Spacelab Operations by employing planning margins and enhanced use of telescience capabilities. The operational concept is based on the following responsibility distribution:

- Experiment Operations: Prime Investigator, UHB
- Payload Facility coordination: USOC
- European Payload operations coordination vis-à-vis POIC: MSCC

Generally, the MSCC will be able to interface with up to 9 User Support and Operations Centers (USOC) and 15 to 20 User Home Bases (UHB).

Interfaces to user sites will ultimately be tailored to each user needs and the total extent of user communications services will probably vary from one increment to the next. Therefore, interfaces between the MSCC and users must rely upon a generalization of user communication needs until that time when actual users are available.

Services provided by the MSCC to the Users are:

- Operations Preparation
 - European Execution level planning and replanning by integration of harmonized Requests from USOCs
- Operations Execution
 - European Payload resource monitoring and control
 - European Payload health and safety control
 - Payload command coordination/validation incl. telescience throughput coordination
 - End-to-end communication coordination
- User Data Services
 - User Data Service Coordination
 - Data Quality Monitoring
 - Data Accountability
 - Intermediate Data Recording and Data Product Generation / Distribution

In selected cases, users may also be accommodated in MSCC internal user rooms during their operations "campaigns". In such cases, the user interfaces will be treated identically to those located externally.

MAKING ADAPTABLE SYSTEMS WORK FOR MISSION OPERATIONS: A CASE STUDY

N94-23883

Barbara E. Holder
Michael E. Levesque

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr.
Mail Stop 171-243
Pasadena, CA 91109
USA
barbara@jpl-devvax.jpl.nasa.gov
mikel@jpl-devvax.jpl.nasa.gov

ABSTRACT

The Advanced Multimission Operations System (AMMOS) at NASA's Jet Propulsion Laboratory is based on a highly adaptable multimission ground data system (MGDS) for mission operations. The goal for MGDS is to support current flight project science and engineering personnel and to meet the demands of future missions while reducing associated operations and software development costs. MGDS has become a powerful and flexible mission operations system by using a network of heterogeneous workstations, emerging open system standards, and selecting an adaptable tools-based architecture. This paper describes challenges in developing adaptable systems for mission operations and the benefits of this approach.

Key Words: User-configurable systems, ground data systems, mission operations.

1. INTRODUCTION

The Advanced Multimission Operations System (AMMOS) at NASA's Jet Propulsion Laboratory is based on a multimission ground data system (MGDS) that is adaptable to individual flight projects and thus eliminates the need for each flight project to build its own separate ground data system. MGDS goals [1] are to:

- Support each new mission with all of its mission-unique processing, as a ready

adaptation to a baseline set of functional capabilities.

- Simplify data distribution and access services so analysts can readily locate and analyze their data using MGDS.
- Maximize multimission practices, in order to reduce training and operational costs, by using similar equipment and user interfaces across functions and missions.
- Support operations for 10-15 years, by evolving the MGDS in response to hardware improvements and changing mission needs or types.

The MGDS architecture is a set of layers of individual components that are added, replaced, or adapted, as technologies, and mission scenarios change [2].

The layers, shown in Figure 1, are:

- Applications
- Organization specific standards
- Industry standards
- Open system standards.

The components within the layers are the system's building blocks: libraries, tools, tables, scripts, and operations scenarios that may be reused by different flight projects and software developers to reduce development and maintenance costs. Most functions are provided via the tools [Figure 2]. A tool is software easily configurable to become an application with utility to users. The tools themselves are rarely modified, but by modifying scripts, tables, and operational

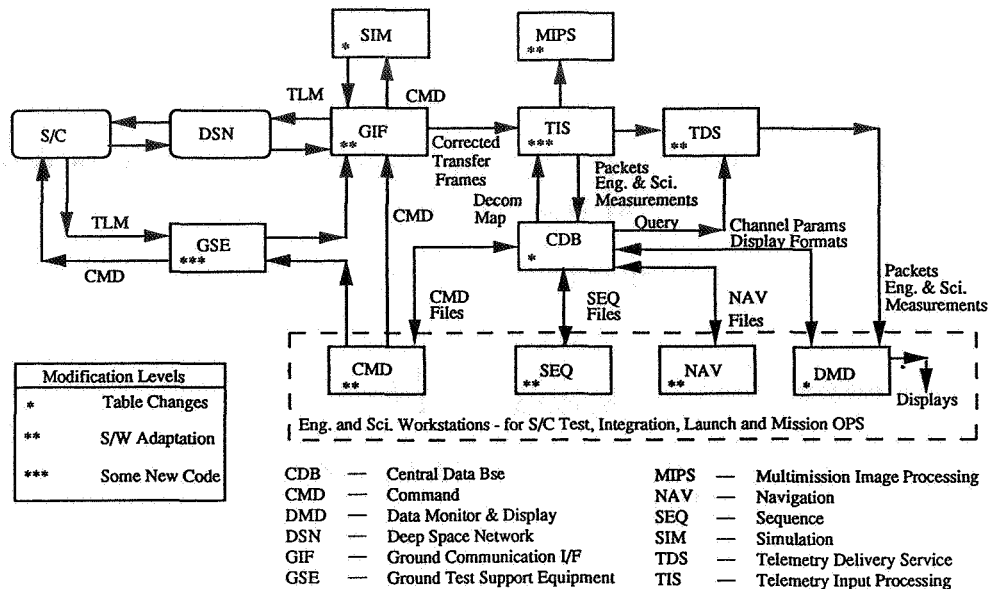


Figure 2. AMMOS Tool Suite

A system based on adaptable tools delivers high capability per development dollar, because the users evolve the system by modifying the scripts and tables to suit their needs. As a result, users discover highly efficient methods for accomplishing their tasks that were never anticipated by the developers or managers.

For example, the Magellan spacecraft team regularly did a variety of tasks on a daily, weekly, and one-time basis. The users automated repetitive tasks by writing scripts that performed the task for them. Analysts could then spend more time doing real work: mission analysis and anomaly resolution.

3.2 The Turn-key Example

Despite the success of Magellan's adaptation, it was clear that MGDS lacked the usability of a turn-key system. Many project managers and users thought they should not have to be involved in project adaptation. There was a discrepancy between what the developers gave the projects (an adaptable system) and what the projects were expecting (a turn-key system). To resolve this conflict, the development organization added a turn-key system delivery capability:

- adaptation teams as discussed earlier;

- a system user interface for novice users, based on the X.Desktop software package.

The system user interface (SUI) enabled novice users to become familiar with the tools and UNIX in a simplified user environment. However, once they became more familiar with MGDS and UNIX, they often abandoned SUI. MGDS's turn-key environment proved invaluable for novice users, but not for advanced users.

4. ADAPTATION DETAILS

4.1 System Adaptation

To meet MGDS's design goals of supporting operations for the next 10-15 years and evolving with hardware improvements and changing mission needs or types, MGDS was built on open system technology so it could be portable to other hardware and software environments. Figure 1 illustrates the open system layering [6] used in the MGDS design. Application programs (the tools) are not forced to change when new hardware and operating system software environments change.

As open system standards evolve, they will be incorporated into MGDS, and less capability will need to be provided by

scenarios the tools can be modified to meet a mission's specific needs.

MGDS users may customize their work environment themselves or with the assistance of a developer. The system evolves into an efficient mission operations environment when users are empowered to modify the building blocks or combine them in various ways to meet their mission-unique needs.

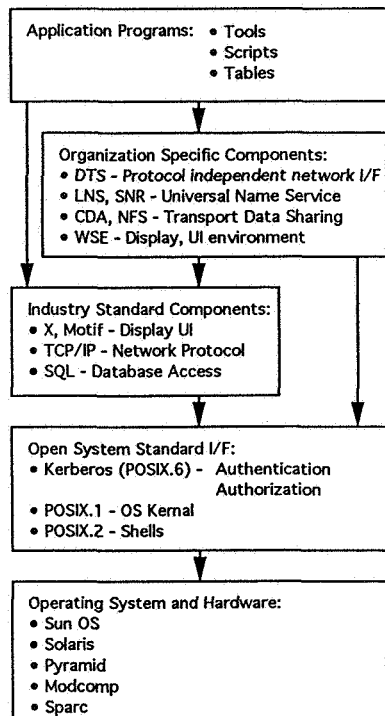


Figure 1. Open System Layer Model

2. MISSION OPERATIONS

MGDS is a suite of adaptable tools that support functions common to all missions. Each tool supports a generic mission operations function that may be needed by any mission.

When the tool suite is delivered to a flight project, it is adapted from the generic version to a project version. The adaptation process is provided as a service to the flight projects by the mission operations program office. The initial project adaptation is made by software developers and integrators, to meet project requirements and scenarios. The

second adaptation is made by either a handyman user or an adaptation team comprising users and programmers; to meet general end-user and flight team needs. The final adaptation is made by the end-users who may further adapt the tools to meet their individual or team requirements.

An adaptation team creates an atmosphere of cooperation between the development organization and the operations organization. The adaptation team concept is described by MacLean et al [4] as a *handyman user* who facilitates communication between operators and developers, and responds to the users' immediate needs. The adaptation team or handyman user communicates user needs to programmers for longer-term or more complex development.

3. HOW MGDS IS ADAPTED

3.1 The Handyman Example

MGDS functions are normally made available to projects as adaptable tools [Figure 2]. Users decide how to use the tools to do their tasks.

For example, the capability to process and display decommutated telemetry is provided by the Data Monitor and Display (DMD) tool. The Magellan spacecraft team used DMD to monitor real-time telemetry via the spacecraft team workstations [4]. Each spacecraft subsystem team had a DMD environment that was adapted to subsystem requirements and operator needs by a handyman user. Each spacecraft subsystem had different data display formats, processed different channels, and needed different scripts to alter the processing and display of the data.

Engineers defined their own display screens or templates by using a special template definition language (TDL). Although TDL is limited, it provided users with the tools required to create their own imaginative displays. Users could also produce derived telemetry channels via the channel conversion language. The MGN flight team developed 250 derived channels on their own.

organization-specific components. However, even with organization-specific standard components, as application programs become more numerous, or are enhanced, the open systems layered model will provide the framework for increased portability.

4.2 Primary Project Adaptation

The initial project adaptation is made by software developers and system integrators.

The developers ensure that the functional requirements of the project are met in the delivered tool suite. For instance, a developer modifies the DMD to process mission specific data types as part of the initial adaptation. Since these data types follow a baseline data format standard, the required changes are easily made by updating the DMD tables. Unanticipated adaptations will increase development costs and make the project adaptation more difficult.

The integrators test tools by writing scripts that set up end to end data processing to emulate mission scenarios.

The adaptable system design makes software development and integration easy because tools are built on common system components that rarely change.

4.3 Secondary Adaptation for Users and Teams

We found the adaptation team or handyman approach worked better than the turn-key approach for meeting mission operations requirements. The success of the adaptation rests on how well the team or handyman supports the users. Most of the handyman time must be spent in the missions support area, working with users at their workstations so that user environments are configured on user request with immediate feedback from users. Adaptation engineers are then able to inform developers of user problems, new mission requirements, and user requests that must be implemented in the future.

4.4 Is That It?

Designers will always miss something. That's why adaptable systems really succeed at the user level. Once the initial and secondary adaptations are made, the users will evolve the system into its most workable configuration for operations.

User adaptations can be as simple as grouping a set of tool commands into a script or as complex as writing a UNIX shell script to process data through several tools via named pipes. Although the users can make the system work for their tasks, they do not always choose the most productive methods [5]. Still, users often change it in ways that were never anticipated by the developers, integrators or adaptation engineers.

4.5 Adaptation is a Continuous Process

Managers must encourage a cooperative development and operational environment and define tangible system goals and benchmarks. Creating and maintaining a culture for adaptation is a major component of adaptable system development. Developers and users must all accept responsibility for the success of the adaptation. Feedback and interaction among developer, tester, and operator is indispensable. System teamwork is the key to solutions that are based on a better understanding of system-wide tradeoffs.

5. LOOKING AHEAD

The adaptable nature of MGDS will readily support new missions and operational scenarios. New missions will be able to use the tools provided by MGDS as part of their low-cost plan. Adaptations are more cost effective if missions consider and adopt MGDS baseline capabilities and standards prior to defining mission unique requirements.

5.1 Lessons Learned

- The handyman approach works better than the turn-key approach.

•The MGDS design is a good approach for creating a powerful, flexible, open system for mission operations.

•The adaptation process is only as good as the communication between operations and development organizations. Management must encourage teamwork and provide a structure for making system tradeoffs.

•Adaptable systems work best with experienced users who are eager to migrate the system into optimal mission operations. The users are the key to success and should not be automated out of the system. Rather, the users should automate the system to better serve them and science.

6. ACKNOWLEDGMENTS

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration. The authors would like to thank Scott Burleigh, Steve Jenkins and Mike Tankenson for their valuable comments on earlier drafts of this paper and Robyn Lefler for producing the figures.

7. REFERENCES

1. SFOC¹ System Engineering, SFOC System Functional Design Document. *JPL Internal, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109*, November 1986.
2. A.W. Bucher. Using SFOC to Fly the Magellan Venus Mapping Mission. *Magellan Forum Series, JPL Internal, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109*, July 1992.
3. A.J. Gainsborough, J.A. Holladay, P.H. Ray. A Distributed Computer Environment for a Multimission Spacecraft Operations and

Control Center. *Proceedings of ESA Symposium: Ground Data Systems for Spacecraft Control*, Darnstadt, FRG, 26-29, June 1990.

4. A. MacLean, K. Carter, L. Lovstrand and T. Moran. User-Tailorable Systems: Pressing the Issues with Buttons. *Human Factors in Computing Systems, CHI'90 Conference Proceedings (Seattle, WA)*, ACM, New York, 1990, pp. 175-182.

5. E. Nilsen, H. Jong. Method Engineering: From Data to Model to Practice. *Human Factors in Computing Systems, CHI '92 Conference Proceedings (Monterey, CA)*, ACM, New York, 1992, pp. 313-319.

6. Technical Committee on Operating Systems and Application Environments of the IEEE Computer Society. Standards Project Draft Guide to the POSIX Open Systems Environment, P1003.0. *Institute of Electrical and Electronics Engineers, Inc.* New York, 1991.

¹ The Advanced Multimission Operations System (AMMOS) was formerly named the Space Flight Operations Center (SFOC).

N94-23884

AN APPROACH TO THE DESIGN OF OPERATIONS SYSTEMS

Roy L. Chafin and Patrick S. Curran
Space Flight Operations Section
NASA Jet Propulsion Laboratory
4800 Oak Grove Drive
Pasadena, Calif. 91106
818/354-325
818/393-4110 (FAX)

1. BACKGROUND

The MultiMission Control Team (MMCT) consists of mission controllers which provides Real-Time operations support for the Mars Observer project. The Real-Time Operations task is to insure the integrity of the ground data system, to insure that the configuration is correct to support the mission, and to monitor the spacecraft for the Spacecraft Team. Operations systems are typically developed by adapting operations systems from previous projects. Problems tend to be solved empirically when they are either anticipated or observed in testing. This development method has worked in the past when time was available for extensive Ops testing. In the present NASA budget environment a more cost conscious design approach has become necessary. Cost is a concern because operations is an ongoing, continuous activity. Reducing costs entails reducing staff. Reducing staffing levels potentially increases the risk of mission failure. Therefore, keeping track of the risk level is necessary.

2. INTRODUCTION

The role of the MMCT is to interact with the process (Mars Observer mission) to accomplish required tasks. The organizations design discussed here is to develop an organization of people, equipment, software, and procedures that will accomplish these tasks. The goal is to provide a design technique that can produce an operations organization that will meet the requirements placed on it, with minimum costs and with the understanding of the risks involved.

The design approach is based on considering the Mars Observer mission as a process. The Mars Observer mission is a rather linear process. The spacecraft is launched; then, it goes through a well specified sequence of actions until the end of the mission.

The following Operations System design approach was developed for the design of the MMCT to support the Mars Observer mission.

The design technique consists of:

Identifying the Mars Observer Mission process.

Modeling the process.

Identifying the requirements imposed by the Mars Observer Project on the MMCT.

Synthesizing the MMCT scenarios that respond to the mission requirements, both imposed and implied requirements.

Derive requirements for support from other parts of the operations organization.

Analyzing scenarios for staffing requirements, training requirements, workload problems, etc.

Reviewing the imposed requirements from the Project for feasibility. Requirements that cannot be accommodated are negotiated.

Developing staffing plans, training plans, test plans, etc.

The Mission sequence model is documented in the MMCT Design Document. The Project requirements and the MMCT operating scenarios are integrated into the design document.

This approach to the design of the MMCT provides a more complete understanding of the mission processes and a cost effective method of the tailoring of the MMCT operations system to support the Mars Observer mission. The purpose of this paper is to present this approach and discuss its merits.

3. APPROACH

The Mars Observer mission process is identified from the Mars Observer Mission Sequence Plan (Ref. 1). This document identifies the spacecraft activities that are to be supported by the MMCT. The Mars Observer Mission Operations specification Volume 3: Operations (Ref. 2) present the requirements that the MMCT must meet to support the project.

The mission sequence is modeled in a form that allows for hierarchic refinements. To facilitate this effort, the commercial computer program SDDL was chosen. SDDL is a Pseudo English language intended for software program design. The Mars Observer Mission Sequence Plan was the basis for decomposing the mission process from an overall description through subprocesses to elementary processes. Typically, these elementary processes were sufficiently simple to be described on a single page. SDDL provided the capability to reference subprocesses through CALL statements in the manner of a software subroutine. Figures 1 and 2 illustrate the decomposition of the mission process.

Verification of the process model is provided by joint reviews between the Mars Observer MMCT design team and the Spacecraft Control Team (SCT). The requirements presented by the Mars Observer Project are analyzed in terms of their impact on the Mars Observer MMCT organization system. The requirements are clarified so that they are consistent for both the originating and the responding organizations. The imposed requirements are integrated into the

design document where they apply. SDDL has the capability of indexing the requirements and placing the index at the end of the design document. Figure 3 illustrates the requirements and the requirements index.

MMCT operations scenarios are developed to accomplish the required tasks. They are written into the Mars Observer process model to form the MMCT Design Document. Scenarios that satisfy the imposed requirements are integrated with the requirements to provide requirements traceability. Operations scenarios are illustrated in Figures 2 and 3.

The requirements are then negotiated between the Mars Observer MMCT and the Mars Observer Project. Requirements are accepted, waived, or when problems exist workaround solutions are identified. The requirements are refined and documented in the design document.

The operating scenarios are reviewed by experienced mission controllers. Experience from prior missions is used to test the validity of the scenarios. A person with actual experience usually can tell whether a task (scenario) can be accomplished in the time required and with the resources allowed.

From the Mission Controller Team scenarios, the resources required to support the Mars Observer mission are identified. These resources include staffing, data, hardware, software, work-station displays, procedures, logs, reports, and management interactions. Displays that are required to support specific MMCT tasks are identified, specified, and indexed in the design document.

Derived requirements identified above are placed in the Design Document at their point of application and again are indexed with the SDDL indexing capability. Derived requirements are illustrated in Figures 2 and 4. Derived requirements are requirements that are derived from the exposition of the operating scenarios. They are the data, procedures, equipments, support, etc. that are recognized as needed to accomplish the required MMCT tasks.

Discrepancies discovered in the process of developing and analyzing scenarios are recorded as unresolved issues. Unresolved issues are identified and indexed. This allows unresolved issues to be tracked. The unresolved issues index is illustrated in Figure 5.

Scenarios are analyzed and workload studies are performed. These workload studies are used to identify when controllers are overloaded. They also identify when one controller may be available for additional mission responsibilities, thereby improving multimission operation.

The detailed workload studies and requirements analyses indicate when a specific spacecraft sequence overloads the mission controller, or when resources are not adequate to support the mission operations. This provides an understanding of specific risks of failure. It provides the basis for the MMCT development team to negotiate additional staffing, specific workstation displays, software tools, data validation programs, additional spacecraft or mission information, or if required additional time to accomplish specific tasks.

The MMCT Design Document then provides the basis for staffing and training plans. The design document provides the basis for determining whether the mission controller will be operating the Knowledge based mode or the Procedural based mode.

One of the basic parameters of designing an operations system is whether the operation will be Knowledge based or Procedural based. That is, will the normal operation be based on the operators knowledge of the process or will the operator normally be guided by preplanned procedures. The advantage of Procedural based operations design is that the skill requirements on the operator is less than for a Knowledge based operations system. We can expect the operator costs to be less for Procedural based system than for Knowledge based system. Procedural based system design can be used when the basic process is well known and relatively simple (i.e. procedures can be written), and the

basic system is stable (i.e. procedures are continuously valid).

When the basic system process is not well understood or the process changes, adequate procedures are difficult, therefore the system must be operated as a Knowledge based system. This requires that the operator be sufficiently knowledgeable of the system process that he can recognize when problems occur and can formulate plans to resolve the problems. The advantage of a Knowledge based system is that preplanning is minimized, and the operator responds to problems when they occur. If a Procedural based operation is appropriate, then the necessary procedures are identified and plans for developing them are generated. If a Knowledge based operations is more appropriate, then the necessary training plans are identified and developed.

4. CONCLUSION

The Mars Observer MMCT Design Document, as presented in the SDDL format, serves as the repository for the Mars Observer mission process model, the imposed requirements, the synthesized Mars Observer Controller responsibilities, the derived requirements, and unresolved issues.

The Mars Observer MMCT Design Document provides the basis for developing operations procedures, staffing plans, and training plans.

The Mars Observer MMCT Design Document provides a clear basis for the negotiation of resources with other organizations. It also provides the tracking of derived requirements and unresolved issues. It provides a tool for working out the details of the implementation. It provides the structure on which the details of the operations scenarios are analyzed to uncover problems and inconsistencies.

The design techniques presented for the MMCT Operations design provide a clear, rational, cost effective design process.

With a better understanding of the Operations System development come better cost control and risk management.

5. REFERENCES

1. Mars Observer Mission Sequence Plan, Vol 1: Mission Sequencing Scenarios, 642-313, Vol. 1. November, 1991. Jet Propulsion Laboratory, JPL D-3826, Vol 1.

2. Mars Observer Mission Operations Specifications, Vol 3: Operations Encounter Version, 642-315, Vol 3. September, 1991. Jet Propulsion Laboratory, JPL D-3822.


```

84 PROGRAM Mission Phase
85 *****
86 * This module is the top level of the mission activity hierarchy. *
87 *****
88
89 +-----+
90 + At any point in the MO mission activity the MCT has +
91 + the capability to: +
92 + * Transmit required commands to spacecraft +
93 + * Verify spacecraft receipt of commands +
94 + * Identify GDS conditions that interrupt or +
95 + degrade command transmissions +
96 + * Assure continued acquisition, safeing of required data +
97 + * Verify accomplishment of the SOE +
98 + * Identify unexpected interruptions or degradations +
99 + of the required data +
100 + * Initiate troubleshooting procedures when data product, +
101 + command interruptions, degradations occur +
102 + * Coordinate GDS recovery from data product and +
103 + command interruptions and degradation +
104 + * Develop, analyze real-time S/C, GDS trends +
105 + * Report observed spacecraft data anomalies to SCT +
106 + * Respond to and coordinate real-time SOE changes +
107 +-----+
108
109 + Note: A success-oriented mission activity is assumed in the +
110 + following analysis +
111 +-----+
112 SELECT Mission Phase
113
114 CASE Launch
115 CALL Launch_Phase----->( 3)
116
117 CASE Inner Cruise
118 CALL Inner_Cruise----->( 4) ---->
119
120 CASE Outer Cruise
121 CALL Outer_Cruise----->( 5)
122
123 CASE Orbit Insertion
124 CALL Orbit_Insertion----->( 6)
125
126 CASE Mapping
127 CALL Mapping----->( 7)
128
129
130
131
132
133
134 ENI

```

PAGE 4

```

142 PROGRAM Inner_Cruise
143 *****
144 * Mission activities in the Inner Cruise Phase *
145 *****
146
147 SELECT Inner cruise phase sequence
148
149 CASE Spacecraft checkout (C1)
150 CALL Spacecraft_Checkout----->( 8)
151
152 CASE TCM 1 (C2)
153 CALL TCM_1----->( 23)
154
155 CASE Payload Checkout (C3)
156 CALL Payload_Checkout----->( 25)

```

PAGE 8

```

192 PROGRAM Spacecraft Checkout
193 *****
194 * ID C1, Duration 13 days, Start I+2 days, End I+15 days *
195 * Continuous DSN coverage *
196 *****
197
198 !-----!
199 ! ASSUMPTIONS !
200 ! The SCT is on duty !
201 !-----!
202
203
204 LOOP for duration of C1
205
206 CALL Monitor----->( 55)
207
208 Do the C1 activities as they are required
209 CALL C1_Activities----->( 9) ----> Figure 2
210
211 IF Shift change
212 CALL Shift_Change----->( 48)
213 ENDIF
214
215 IF Station transfer
216 CALL Station_Transfer----->( 51)
217 ENDIF
218
219 ENDLOOP
220
221 Return to
222 CALL Inner_Cruise----->( 4)
223

```

Figure 1

```

226 PROGRAM C1_Activities
227   SELECT
228     CASE Launch playback, load A
229       Send command load C1A (I+1/12:00:00)
230
231       CALL Commanding----->( 46)
232       CALL C1A_Activities----->( 11)
233
234     CASE REDMAN load
235       Send command REDMAN load
236       CALL Commanding----->( 46)
237       CALL REDMAN_Activities----->( 13)
238
239     CASE Propulsion priming, MAG & GRS boom extensions, load B
240       Send command load C1B (I+3/12:00:00)
241       CALL Commanding----->( 46)
242       CALL C1B_Activities----->( 14)
243
244     CASE PDS health check & GRS calibration data, load C
245       ++++++
246       + If any anomaly or other difficulty is encountered during +
247       + the spacecraft checkout, this sequence load may be omitted +
248       + to allow more time for the ground to interact with +
249       + the spacecraft. +
250       ++++++
251       Send command load C1C (I+5/12:00:00)
252
253       CALL Commanding----->( 46)
254       CALL C1C_Activities----->( 17)
255
256     CASE Tank Pressurization, load D
257       Send command load C1D to open valve 7
258       (I+11/12:00:00)
259       CALL Commanding----->( 46)
260       CALL C1D_Activities----->( 18)
261
262     CASE Tank Pressurization, Load E
263       IF valve 7 was successfully opened
264         Send command load C1-E(P) to open valve 5 (I+13/04:00:00)
265         CALL Commanding----->( 46)
266         CALL C1EP_Activities----->( 20)
267       ELSE
268         Send command load C1-E(B) to open
269         both valves 5 and 8 (I+13/04:00:00)
270         CALL Commanding----->( 46)
271         CALL C1EB_Activities----->( 21)
272       ENDIF
273
274     CASE Tank Pressurization, Load F
275       IF valve 5 was not successfully opened
276         Send command load C1F to open valve 6 (I+13/12:00:00)
277         CALL Commanding----->( 46)
278         CALL C1F_Activities----->( 22)
279       ENDIF
280   ENDSELECT
281
282

```

PAGE 11

```

288 PROGRAM C1A_Activities
289 *****
290 * This is the scenario for the mission controller to handle *
291 * the C1A activities *
292 *****
293
Derived | C1A activities scenario |
Requirement [IN.01.1 C1A activities scenario should be reviewed with SCT]

----> Callup the -C1A display-
      'D.3.1 C1A display'

300 Confirm USO selected (LQ013/L0020) (I+2/16:00:00)
301 Confirm Ranging enabled (L0009/L0016) (I+2/16:00:00)
302 Confirm RPA 2 filament is off (L0029) (I+2/16:00:30)
303 Callup -DTR display-
304 Confirm that DTR1 is active (C0016) (I+2/16:01:00)
305 Confirm Sun Monitor disabled (?) (I+2/16:02:00)
306 [IN.01.2 What is the Sun Monitor disabled channel number] <--- Unresolved Issue
307
308 Conf long-term gyro recovery enabled (F4064) (I+2/16:03:00)
309
310 Confirm new battery charge rate (I+2/16:04:00)
311 Charge rate 1: E0501 2: E0503
312 Voltage limits 1: E0301(H)
313 2: E0303(H)
314
315 Playback DTR3 at 8 kbps (I+2/16:30:00)
316
317 CALL DTR_Playback----->( 41)
318 Playback DTR2 at 32 kbps (I+2/20:00:00)
319
320 CALL DTR_Playback----->( 41)
321 Return to 2 kbps ENG telemetry (I+2/23:00:30)
322

```

Figure 2

```

1891 PROGRAM Station_Data_Recall
1892
1893 *****
1894 * The following procedure is used to recall recorded data *
1895 * from the station that for some reason did not get into the PDB. *
1896 *****
1897 "R 5.2.2.1.7 Telemetry data gaps, playback" <----- Requirement
1898
1899 | Telemetry data playback from DSCC scenario |
1900 | IN.29.1 TLM data playback from DSCC scenario needs validation |
1901
1902 Request DSCC to playback required telemetry data
1903
1904 Configure SFOC for DSCC data playback
1905
1906 Confirm DSCC playback
1907
1908 Confirm telemetry playback data received at PDB
1909
1910 Log telemetry playback
1911
1912 Return to
1913 CALL Outgoing_Station----->( 52)
1914
1915
1916 ENDPROGRAM

```

MMCT Scenario

PAGE 56

```

1979 PROGRAM Imposed_Requirements
1980
1981 *****
1982 * All the imposed requirements are obtained from the *
1983 * Mission Operations Specification Volume 3: Operations *
1984 *****
1985
1986 "R 2.3.1.3 Spacecraft and Ground Health Monitoring"
1987
1988 The FPSO/MCT shall monitor the spacecraft and GDS data when
1989 provided with valid spacecraft/ground predicts, standards,
1990 and limits. The following monitoring scenario supports
1991 this requirement.
1992 CALL Monitor----->( 55)
1993
1994 "R 5.2.2.1.7 Telemetry data gaps, playback"
1995
1996 In the event of telemetry data gaps that need to be filled to
1997 meet Project requirements, the MCT shall coordinate with DSOT
1998 and DAT to assure that telemetry data is recalled from the GIF
1999 or DSCC as soon as possible after the end of the tracking pass
2000 but not to exceed 12 hours.
2001 CALL Station_Data_Recall----->( 53)
2002
2003 ENDPROGRAM

```

Requirements Definition

Imposed Requirements CROSS REFERENCE LISTING

PAGE 71

```

+++++
R 2.3.1.3 Spacecraft and Ground Health Monitoring
PAGE 56 PROGRAM Imposed_Requirements 1986
R 5.2.1.3
PAGE 61 PROGRAM Unallocated_Requirements 2213
R 5.2.1.4
PAGE 61 PROGRAM Unallocated_Requirements 2206
R 5.2.1.6.1
PAGE 61 PROGRAM Unallocated_Requirements 2218
R 5.2.1.6.2
PAGE 61 PROGRAM Unallocated_Requirements 2233
R 5.2.1.6.3
PAGE 61 PROGRAM Unallocated_Requirements 2239
R 5.2.1.7.2
PAGE 61 PROGRAM Unallocated_Requirements 2245
R 5.2.1.7.3
PAGE 61 PROGRAM Unallocated_Requirements 2252
R 5.2.2.1.1.1
PAGE 62 PROGRAM Unallocated_Requirements 2259
R 5.2.2.1.1.2
PAGE 62 PROGRAM Unallocated_Requirements 2264
R 5.2.2.1.1.3
PAGE 62 PROGRAM Unallocated_Requirements 2270
R 5.2.2.1.5.2
PAGE 62 PROGRAM Unallocated_Requirements 2275
R 5.2.2.1.6
PAGE 62 PROGRAM Unallocated_Requirements 2281
R 5.2.2.1.7
PAGE 62 PROGRAM Unallocated_Requirements 2291
R 5.2.2.1.7 Telemetry data gaps, playback
PAGE 53 PROGRAM Station_Data_Recall 1898
PAGE 56 PROGRAM Imposed_Requirements 1994

```

----- Requirements Index

Figure 3

	2007	PROGRAM Derived_Requirements	
	2008		
	2009	'D.1 A top level display is required'	
	2010		
	2011	This display provides a GO/NOGO indication of the configuration	
	2012	and performance of each of the S/C and GDS system.	
	2013	CALL Monitor----->	(55)
	2014		
	2015	'D.2.1 DTR playback display'	
	2016		
	2017	A display is required to support the DTR activities scenarios.	
	2018	CALL DTR_Playback----->	(41)
	2019	CALL DTR_Repack----->	(42)
	2020	CALL DTR_End_of_Record----->	(43)
	2021		

		'D.2.4 Spacecraft Maneuver display'	
		A display is required to support the spacecraft maneuver	
		scenario.	
	2025	CALL Maneuver----->	(24)
	2026		
	2027	'D.3.1 C1A display'	
	2028		
	2029	A display is required to support the C1A mission segment.	
	2030	CALL C1A_Activities----->	(11)
	2031	CALL REDMAN_Activities----->	(13)
	2032		
	2033		
	2034	'D.3.2 C1B display'	
	2035		
	2036	A display is required to support the C1B mission segment.	
	2037	CALL C1B_Activities----->	(14)
	2038		
	2039	'D.3.3 C1C display'	
	2040		
	2041	A display is required to support the C1C mission segment.	
	2042	CALL C1C_Activities----->	(17)
	2043		
	2044	'D.3.4 C1D display'	
	2045		
	2046	A display is required to support the C1D mission segment.	
	2047	CALL C1D_Activities----->	(18)
	2048	CALL C1EP_Activities----->	(20)
	2049	CALL C1EB_Activities----->	(21)
	2050	CALL C1F_Activities----->	(22)
	2051		
	2052	'D.3.6 C3A display'	
	2053		
	2054	A display is required to support the C3A mission segment.	
	2055		
	2056	CALL C3A----->	(26)
	2057		

Derived Requirements

Definition

Derived Requirements
CROSS REFERENCE LISTING

PAGE 72

D.1 A top level display is required			
PAGE 57	PROGRAM Derived_Requirements	2009	
D.2.1 DTR display			
PAGE 41	PROGRAM DTR_Playback	1522	
PAGE 42	PROGRAM DTR_Repack	1570	
PAGE 43	PROGRAM DTR_End_of_Record	1629	
D.2.1 DTR playback display			
PAGE 57	PROGRAM Derived_Requirements	2015	
D.2.4 Spacecraft Maneuver display			
PAGE 57	PROGRAM Derived_Requirements	2022	
D.2.5 Spacecraft Maneuver display			
PAGE 24	PROGRAM Maneuver	822	
D.3.1 C1A display			
PAGE 11	PROGRAM C1A_Activities		
PAGE 57	PROGRAM Derived_Requirements		
D.3.10 C5 display			
PAGE 34	PROGRAM Outer_Cruise_Transition	1269	
PAGE 58	PROGRAM Derived_Requirements	2075	
D.3.2 C1B display			
PAGE 14	PROGRAM C1B_Activities	403	
PAGE 57	PROGRAM Derived_Requirements	2034	
D.3.3 C1C display			
PAGE 17	PROGRAM C1C_Activities	509	
PAGE 57	PROGRAM Derived_Requirements	2039	
D.3.4 C1D display			
PAGE 18	PROGRAM C1D_Activities	543	
PAGE 57	PROGRAM Derived_Requirements	2044	

Derived Requirements Index

Figure 4

+++++

IN.01.1 C1A activities scenario should be reviewed with SCT		
PAGE 11 PROGRAM C1A Activities	345	
IN.01.2 What is the Sun Monitor disabled channel number		Unresolved Issue Index
PAGE 11 PROGRAM C1A Activities		
IN.01.4 How to confirm that heads are parked		
PAGE 11 PROGRAM C1A Activities	345	
IN.02.1 REDMAN activities scenario should be reviewed with SCT		
PAGE 13 PROGRAM REDMAN Activities	360	
IN.02.2 What is the battery temp cont channel number		
PAGE 13 PROGRAM REDMAN Activities	364	
IN.02.3 What is the battery charge contr channel number		
PAGE 13 PROGRAM REDMAN Activities	367	
IN.02.4 Is E0001 the correct CN for battery backup charge?		
PAGE 13 PROGRAM REDMAN Activities	371	
IN.02.5 How is the contingency alert enable confirmed		
PAGE 13 PROGRAM REDMAN Activities	374	
IN.02.6 How is the telemetry Verification enable confirmed		
PAGE 13 PROGRAM REDMAN Activities	377	
IN.02.8 What parameters and thresholds are used in REDMAN		
PAGE 13 PROGRAM REDMAN Activities	386	
IN.03.1 C1B activities scenario needs validation		
PAGE 14 PROGRAM C1B Activities	400	
IN.03.2 How is Biprop system vented and primed confirmed		
PAGE 14 PROGRAM C1B Activities	411	
IN.03.3 Do we need to look at the HGA GDE on signal?		
PAGE 14 PROGRAM C1B Activities	416	
IN.05.1 C1C activities scenario should be reviewed with SCT		
PAGE 17 PROGRAM C1C Activities	506	
IN.06.1 C1D activities scenario should be reviewed with SCT		
PAGE 18 PROGRAM C1D Activities	540	
IN.07.1 C1-E(P) activities scenario should be reviewed with SCT		
PAGE 20 PROGRAM C1EP Activities	608	
IN.08.1 C1-E(B) activities scenario should be reviewed with SCT		
PAGE 21 PROGRAM C1EB Activities	662	
IN.09.1 C1-F activities scenario should be reviewed with SCT		
PAGE 22 PROGRAM C1F Activities	718	
IN.10.1 TCM-1 Operational scenario should be reviewed with SCT		
PAGE 23 PROGRAM TCM 1	772	
IN.11.1 Maneuver scenario should be reviewed with SCT		
PAGE 24 PROGRAM Maneuver	817	
IN.11.2 What is MCT responsibility during maneuvers		
PAGE 24 PROGRAM Maneuver	819	
IN.11.3 How do we confirm spacecraft momentum unload		
PAGE 24 PROGRAM Maneuver	825	
IN.11.4 What does MCT do on a faulty maneuver		
PAGE 24 PROGRAM Maneuver	863	
IN.12.1 Payload checkout scenario should be reviewed with SCT		
PAGE 26 PROGRAM C3A	936	
IN.12.2 What is the PDS write protect on/off channel number		
PAGE 26 PROGRAM C3A	949	
IN.12.3 How are three redundant PDS memory readouts confirmed		
PAGE 26 PROGRAM C3A	960	
IN.12.4 How is command PDS to RAM confirmed		
PAGE 26 PROGRAM C3A	963	
IN.12.5 On switch to S&E-1 MBR, is there any effect on GDS		

Figure 5

OBJECT-ORIENTED TECHNOLOGIES IN A MULTI-MISSION DATA SYSTEM

Susan C. Murphy
 Kevin J. Miller
 John J. Louie

N 9 4 - 2 3 8 8 5

Operation Engineering Lab
 Jet Propulsion Laboratory, 301-345
 California Institute of Technology
 Pasadena, California 91109-8099

ABSTRACT

The Operations Engineering Laboratory (OEL) at JPL is developing new technologies that can provide more efficient and productive ways of doing business in flight operations. Over the past three years, we have worked closely with the Multi-Mission Control Team to develop automation tools, providing technology transfer into operations and resulting in substantial cost savings and error reduction. The OEL development philosophy is characterized by object-oriented design, extensive reusability of code, and an iterative development model with active participation of the end users. Through our work, the benefits of object-oriented design have become apparent for use in mission control data systems.

In this paper, we will explain object-oriented technologies and how they can be used in a mission control center to improve efficiency and productivity. We will also discuss the current research and development efforts in the JPL Operations Engineering Laboratory to architect and prototype a new paradigm for mission control operations based on object-oriented concepts.

Key Words: Operations, Automation, Data Analysis, Object-Oriented

1. INTRODUCTION

The Multi-Mission Ground Data System (MGDS) at JPL has brought improvements and new technologies to mission operations. The development of a generic data system to meet the needs of multiple missions was intended to avoid re-inventing capabilities for each new mission and thus reduce costs. The

traditional mainframe-based data systems of the past were expensive to modify and their proprietary architectures did not facilitate incorporation of new technologies. The MGDS is based on a distributed architecture, with powerful UNIX workstations, incorporating standards and open system architectures.

The MGDS is being expanded beyond its data delivery capabilities to include automation and analysis tools for the more demanding missions of the future. However, automation tools can help reduce costs only if they are focused on the people and the tasks they perform. New technologies may only bring minimal cost savings if the new system functions much like the old one. This often happens since the users who write the requirements aren't always familiar with the capabilities of new technologies and simply use their existing system as a model. For example, the mission controllers asked for a scrolling screen that displayed telemetry values representing the latest value of the spacecraft clock. This was the way the old system allowed them to determine whether there were any data outages. The developers gave them their scrolling display and operators continued to stare at these displays watching for outages. An important opportunity was lost to automate this process and improve the efficiency of operations. To solve these types of communications problems, a new approach was tried. Each division was assigned responsibility for its end-to-end system, from development through operations. In response, the Operations Engineering Lab (OEL) was created several years ago to merge operations and development activities for the Space Flight Operations Section.

2. OPERATIONS ENGINEERING LAB

2.1 Development Approach

Over the past several years, the OEL developers have worked closely with mission controllers and spacecraft engineers to develop a set of graphical automation tools. The tools utilize object-oriented graphics and direct-manipulation interfaces. In traditional ground data systems, data presentation and data access are not intuitive. Specialized languages must be learned by the user in order to describe the way data must be processed, accessed, and displayed. An object-oriented approach can simplify the user's interaction with the data system by modeling the system as made up of objects, entities defined by their functional and inherited characteristics. Object-oriented paradigms are ideal for developing easy-to-use graphical user interfaces where data and functions can be activated and manipulated directly on the screen.

Our approach has been successful because we build tools that are integrated, application-specific, and focused on automating essential, yet tedious and time consuming, operations tasks. In addition, we involve users and trainers early in the development process. In fact, we have mission operators work as developers in the lab, sometimes on a part-time basis and, in other cases, full-time for a limited tenure. Conversely, four of the OEL developers worked as members of the Spacecraft and Mission Control Teams at JPL and the Spacecraft Anomaly Team at the Cape in support of the recent Mars Observer launch. This has allowed us to maintain close contact with our users and understand the problems that need to be solved.

We develop software incrementally, as a series of rapid protoflight models that are reviewed constantly by the users. Their active participation has meant that the new technologies have been accepted more readily, and even more importantly, it has often made them enthusiastic to learn a new system. We have found that it is very important to get protoflight implementations in the hands of users and trainers as soon as

possible to provide quick feedback to developers and to gain user acceptance. Our protoflight models have evolved to fully-implemented subsystems of the MGDS that have produced significant cost savings in operations. Similar successes are occurring at other NASA centers as the advances in workstations and open architectures have enabled small programming teams to quickly and cheaply implement automation tools that expand their capabilities [1].

2.2 SEG: An Object-Oriented Development Success Story

One of the initial projects in the OEL was to automate the Sequence of Events Generation (SEG) process that develops the detailed schedules and instructions for the ground control of a spacecraft. The inputs to the SEG process include the spacecraft command sequence, ground resource allocations, and special ground events. The output products include a text listing of the events (the Sequence of Events (SOE)) and a timeline display (the Space Flight Operations Schedule (SFOS)). [2] In the past, much of the SEG process was manual, fragmented, and understood by only a few operators. The mainframe-based software for SOE generation was expensive to maintain and modify and the interface was difficult and usually tedious. The SFOS timeline product was produced on a PC with a word processor. The process was slow, error-prone, and inefficient in the use of personnel. Editing a document was cumbersome, as only a small portion of the page was visible at once. Printing was done on a line printer, and the output was reduced and copied for distribution across the world. The frequent updates to these documents quickly invalidated the user's copy. The hard-copy products made it difficult for users to isolate the events of interest from the thousands of SOE items.

2.3 Building a Multi-Mission SEG

The SEG process was separated into two parts in order to isolate the mission-specific software that was expected to change frequently from the more stable, multi-mission graphical tools that would be needed

to display, edit, view, and print the SOE and SFOS. The mission-specific software are the scripts that automatically generate the ASCII data files that are then input to the SFOS and SOE graphical tools. Both graphical tools were designed to be generic viewing/editing tools with no hard-coded knowledge of the meaning of its contents. As a result, only the scripting software needs to be adapted to produce schedules for different missions and for different application areas. For example, the SFOS editor has been used to produce timeline displays of other schedules such as command sequences and mission planning schedules, simply by changing the file generation scripts.

2.4 Object-Oriented Design

The graphical SFOS editing/viewing tool was designed to make the operator's job fast and easy. The entire SFOS page is visible on the screen, with *What-You-See-Is-What-You-Get* capabilities for viewing, editing, and printing. We used an Object-Oriented (O-O) design approach in which each item can be directly manipulated on-screen as a graphics object.

In an object-oriented design, the system is designed around the **data** that the system must manipulate rather than focusing on the **functions** a system must perform. Objects are defined by their functional characteristics; they encapsulate knowledge of both their current state and expected behavior. Embedded in each object is an 'understanding' of its attributes and the methods it will use for performing its allowed functions. In an O-O system, data objects are often designed to model real-world objects. For example, in the SFOS editor, each object represents a ground or spacecraft event that will occur for a mission. An object that is a spacecraft command belongs to a different class of objects than one that describes a tracking activity.

In the SFOS editor, each object knows how to display, edit, delete, add, and move itself. For example, schedule objects will automatically place themselves in the correct timeline position. When a user selects an object to edit, the object will respond by

bringing up its unique edit dialog box which displays items that are specific to that class of objects. The values displayed in each item reflect the current state values for that particular instance of the object. The menu-driven interface allows a user to change the timeline scale on-the-fly.

The graphics display is saved in an ASCII file which completely describes the format and content of the SFOS. Each line in the ASCII file corresponds to an object on the screen. The standard ASCII file interface was chosen because it allowed the generation of the input data files to be automated using simple scripting languages such as Perl and AWK.

The SOE graphical tool has many of the same features as the SFOS tool. The SOE tool displays a time-ordered, column-formatted display of the SOE file. It has capabilities for searching on selectable criteria, filtering out events of interest, and highlighting events. The contents and format of the columns can be reconfigured by the user.

2.5 Templates for Operations Processes

Although the process was automated, there were many steps in the procedure and decisions had to be made on execution parameters based on a complicated data flow. It was clear that an interface was necessary to simplify the operator's job. An interface builder (OELSHELL) was first implemented for building graphical templates. The template builder was an extension of JPL's D. Smythe's Widget Creation Library which uses a resource file to configure the interface. The templates provide buttons that can be used to call a program and the output can be redirected as needed. File widgets allow a user to designate input files for the process. On-line help and arrows help describe the process.

Templates were built for the SEG process as shown in Figure 1. The SEG template shell allows a user to select which output products are desired and which input sources are needed, eliminating the need for users to know which *program* must be executed in each case. Templates have been built for

other processes and have proved valuable as interfaces for integrating multiple utilities and for use as an operations training tool.

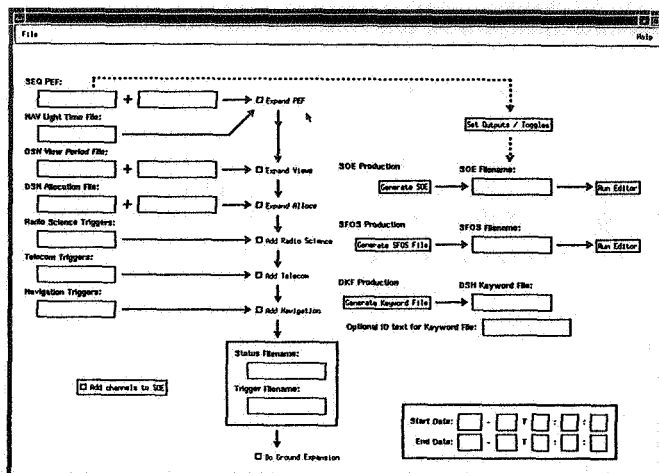


FIGURE 1. SEG TEMPLATE SHELL

The graphical SEG tools have been used as the basis for other editors and operations support tools. The editor/viewers were originally implemented using the SunView windowing environment. With the lab's migration to the new MGDS environment, our software had to be translated into the X/Motif windowing environment. The task was greatly simplified because of the object-oriented approach and the migration was done ahead of schedule and under budget. The system design did not have to change in the new environment, only the user-interaction functions were affected. Our original software design also included a lower-level set of graphics routines for drawing objects on the screen or to a laser printer. Only these routines had to be changed to interface with the X-server. Since all of our software tools used the same graphics routines, our development costs were significantly reduced through this commonality.

2.6 OEL Automation and Analysis Tools

Other automation tools developed in the OEL include an automated telemetry log generator for data management. This on-line log tool has replaced the previous method of hand-written logs describing telemetry data coverage. An alarm clock tool provides user-

programmable clocks and graphics in a workstation window. Another program provides an interactive window-based editing tool for generating milestone schedule charts. A Multi-Mission Smart Alarm tool was built to monitor data outages on MGDS broadcast circuits, eliminating the need for controllers to manually detect outages by watching a scrolling time display. Several other real-time monitoring tools were developed to automatically analyze and log information on engineering packet telemetry, telemetry data gaps, telecommunication uplink frequency verification, and command execution verification. Object-oriented analysis tools have been built, including graphical tools for browsing through telemetry channel information and generating statistical summaries, and plotting channel data. The graphical plotting tool provides sophisticated capabilities for plotting multiple channels versus time or channel versus channel plots. The ASCII file interface has allowed telemetry plots to be produced automatically by generating the input files with simple scripting tools that supply hooks into the MGDS telemetry database retrieval system.

A decommutation map tool provides a graphical means of visualizing the telemetry decommutation process and of interactive editing of a channel object. The tool allows a user to graphically navigate through a decommutation map, moving in and out of sub-commutations using a mouse. The current mode for manipulating decommutation maps consists of editing a file written in a specialized map language. In order to make a change to a specific map, an operator must understand the decommutation map language and must also read through the lengthy source code logic. Undoubtedly, an intuitive, graphical method for visual display and editing of decommutation maps is necessary.

2.7 Results

The software interfaces to the graphical tools are designed to be clean and simple. This has enabled our software to be used for multiple missions with differing objectives. These tools have resulted in substantial savings and reduced errors. The SFOS editor resulted in

a two-thirds reduction in work force required to operate the system and eliminated costly mainframe processing costs. The ASCII file interface and its electronic availability has given scientists access to up-to-date schedules, eliminating printing and shipping expenses that amounted to up to \$200K per project per year. The Smart Alarm Tool has saved up to 40% of the mission controllers time that used to be spent monitoring broadcast lines for data outages. The other automated monitoring tools have eliminated many error-prone, time-consuming manual processes.

It is evident that increased automation of the operations process is necessary and that a careful strategy is required to ensure that the tools focus on specific tasks and that the approach is flexible enough to meet the needs of multiple missions. In addition, the O-O approach has resulted in less costly maintenance, and we have been able to accommodate users' and missions' changing needs with minimal expense.

We have also discovered that developing great tools is not enough. It is essential for developers to get into the operational environment and assist operations teams in adapting the delivered system and its many tools to meet their needs. The OEL has led a Customer Adaptation Team (CAT) for adapting the MGDS for the Voyager and Mars Observer Spacecraft and Mission Control Teams. This effort has been very successful because we work in their environment, configuring the workstations on their desks, building scripts to automate their tasks, and designing interfaces to integrate tools.

3. ADVANCED RESEARCH

3.1 Object-Oriented Operations (O³)

Another emerging trend in the development of data analysis and display software is the use of modular software components (or tools) that are integrated and manipulated by the user as *objects* in a desktop environment. For example, in a ground data system, the software

components would implement operations processes, each with inputs, outputs, and a knowledge base. The objects would closely model the way one thinks about the data system. The objects would understand and account for the rules that govern its behavior and would navigate through the complex layers in the data system itself. The components could be interactively strung together to establish a flow of data according to a user-defined process.

We have proposed an Object-Oriented Operations (O³) system environment that incorporates these techniques in a mission control data system. In a fully-integrated O³ environment, the objects could be invoked by events that occur on the ground or on the spacecraft. For example, if the spacecraft unexpectedly went into safe mode, the ground data system would detect the event and signal appropriate objects (telemetry channels) to be instantiated, producing the necessary displays or analysis of their current status on the screen for visual inspection by the engineer.

In an initial O³ prototype, the OEL is investigating the combination of expert system and object-oriented technologies. Using the Gensym G2 real-time object-oriented expert system, we have built a model of the Mars Observer telecommunications system including the ground system that is driven by the SOE. The spacecraft and each ground station are represented by an icon on a top-level schematic, and the overall state of the system is displayed by changing colors of parts of these icons. Each icon has a specific sub-workspace which can display more specific information about each component of the system. Each object has associated rules that allow the system to compare and verify the state of the spacecraft with respect to the state of the ground tracking stations.

3.2 Object-Based Interaction Paradigm

We are also investigating extending object-oriented concepts to a system environment which provides a mechanism for the tools themselves to communicate through an object-based interaction. The system

interface would allow a user to *drag* objects between components using a mouse. The objects would carry functional information with them, allowing each process to respond to an object in its own unique way. The object-based interaction paradigm has presented some interesting research challenges, and a prototype version has been implemented. In the prototype system, the graphical decommutation map tool and a hypertext dictionary tool interact through common telemetry channel objects. The user can select a channel object in the map and then drag it to the dictionary tool for information on the channel (or vice versa). The object activates different functions in the different tools, enabling a user to focus on the data they want to access and display, rather than on the tools and methods needed to get the job done.

3.3 Closed-Loop Monitoring System

Another OEL research project is an automated closed-loop monitoring system that provides real-time integration of uplink events with downlink telemetry information, using the SOE as the predict source. In the existing SOE for Mars Observer, each spacecraft command item has a descriptive text field that contains a list of related downlink telemetry channels. These channels are monitored in real-time by mission controllers using another tool that reads the downlink telemetry stream. The closed-loop system will integrate these tasks by interfacing the SOE with the real-time telemetry data stream and automatically appending appropriate channel values with command items. To integrate downlink events, the SOE will require a new type of channel data object that is treated as a special field and will be configured to send messages to external processes that will monitor the appropriate channels. Although the channels selected for viewing will be automated, a user may opt to monitor any channel at a given time, simply by adding that channel object at the appropriate point in the time-ordered SOE listing.

This integrated system will greatly simplify the user's ability to access and view telemetry data, and will provide a means to view this data in the context of the commands

and predicted values that are used to interpret it. It also provides a mechanism to automate generation of mission controller logs, *as-flown* SOEs, and dynamic alarms. This tool will also prove very useful in ground support equipment for spacecraft integration and test.

4. CONCLUSION

With new development approaches such as that of JPL's MGDS and Operations Engineering Laboratory, success has been shown in improving mission operability and reducing cost in operations. The use of object-oriented technologies has resulted in software that is easier to use and cheaper to adapt for multiple missions and changing mission objectives. The future of mission control at JPL is one of opportunity and continued improvement. Work at the OEL continues to make use of these opportunities to improve productivity in mission control.

5. ACKNOWLEDGEMENTS

This work was done at the Jet Propulsion Laboratory, California Institute of Technology, under a contract from the National Aeronautics and Space Administration. We would like to acknowledge the work of the technical staff in the OEL, especially Patrick Curran, Roger Davidson, Ana Maria Guerrero, Joseph Hu, Daniel Hurley, Chester Joe, and Christine Aguilera. We would also like to acknowledge the JPL Mission Operations Teams for their enthusiasm and support.

6. REFERENCES

1. Muratore, J., 1991. Real-Time Data System at Space Shuttle Mission Control, *NASA Control Center Conference*.
2. K. Miller and S. Murphy, 1990. Sun Technology for Mission Operations, *Sun Tech Journal*.

MANAGEMENT OF INFORMATION FOR MISSION OPERATIONS USING AUTOMATED KEYWORD REFERENCING

Roger A. Davidson and Patrick S. Curran

N94-23886

Space Flight Operations Section
Jet Propulsion Laboratory, 301-345
California Institute of Technology
Pasadena, California 91109-8099

ABSTRACT

Although millions of dollars have helped to improve the operability and technology of ground data systems for mission operations, almost all mission documentation remains bound in printed volumes. This form of documentation is difficult and time-consuming to use, may be out-of-date, and is usually not cross-referenced with other related volumes of mission documentation. A more effective, automated method of mission information access is needed. In our paper we propose a new method of information management for mission operations using automated keyword referencing. We expound on the justification for and the objectives of this concept. We will share the results of a prototype tool for mission information access that uses a hypertext-like user interface and existing mission documentation. Finally, we will describe the future directions and benefits of our proposed work.

Key Words: Hypermedia, hypertext, information retrieval, mission control, operations, automation

1. INTRODUCTION

Mission complexity, longevity and organizational size have made the documentation of mission information increasingly more important. Unfortunately, documentation is expensive. It must be prepared and distributed. Often, people must be trained in how

the hoards of documents are constructed and used and where they can be found. Then, of course, there are corrections and updates to documents which result in new versions that must be distributed. This says nothing about the configuration control and security issues involved with creation of and access to mission documentation. Moreover, even with all its associated costs and headaches, documentation is worthless unless it can be used effectively. In fact, out-of-date documentation can be misleading, resulting in increased operating cost and risk. One can begin to see why "the good ol' days" of space exploration (when dedicated teams of experts needed less documentation) appear so pleasingly simple.

An example illustrates some of the burdens that hardcopy documentation places on mission operations activities. The Command Dictionary, Telemetry Dictionary, and the Flight and Mission Rules Dictionaries are essential references for the mission operations functions of mission control, command verification, anomaly investigation and training. However, these documents are typically bound separately (having been created by different organizational teams), and the cross-referencing of these related information sets is often found scattered in the textual description of the entries. Therefore an analyst investigating an anomaly on a specific telemetry channel must search for its entry in the telemetry dictionary, read to find related telemetry channels, open the

command dictionary to investigate the commands that affect the telemetry channel, and try to cross-reference with the flight and mission rules to understand the related commanding and scheduling constraints. The resulting, human-generated flurry of bookmarks and Post-it Notes™ is a frustrating, costly and unproductive use of an analyst's time.

1.1 Use of Milo for Galileo

The Jet Propulsion Laboratory's Galileo Project took a significant step towards solving these problems by making their telemetry and command dictionaries and their flight and mission rules available in an electronic form to their mission control and spacecraft teams. In late 1989, an Operations Engineering Team (OET) Systems Engineer named Mark MacDonald wrote a program script in a commercial text editor for personal computers. The program, named Milo, offered only a command line interface, was slow, did not show document graphics, and was not accessible from the principal computer system used by the OET and Mission Control Team (MCT). Even with these negatives, it was observed that OET and MCT personnel were more likely to look for information with Milo than search through piles of printed documents.

1.2 MMIT Prototype

In 1991, JPL's Space Flight Operations Section was funded by the Advanced Systems Office, of the (then) Flight Projects Support Office (now called the Multimission Operations Systems Office), to build a prototype "User Environment for Multimission Control" (Ref. 1). A goal of this work was to prove a new concept called Object-Oriented Operations (O³) in which data objects, representing basic units of mission information such as telemetry channel data, can be shared among different workstation applications using a graphical "drag-and-drop"

user interface paradigm (Ref. 2). For example, a user could be editing a decommutation map with one software tool, "grab" a telemetry channel object then "drag" the object over to an electronic reference tool and "drop" it; at which point the reference tool would display the available mission documentation on that channel.

The approach was to construct three software prototypes that could interact as just described. The chosen tools were a telemetry channel plotting tool, a graphical decommutation map editor, and an electronic reference tool. That reference tool, called the Multi-Mission Information Tool (MMIT), was designed to duplicate all the current capabilities of Galileo's Milo, and incorporate the technologies of graphical user interfaces and hypermedia.

2. CAPABILITIES OF MMIT

MMIT is written in C with the XView™ library and runs in the UNIX environment with the X Window System™. The program can be compiled to run on any Sun™ workstation or compatible. The following paragraphs describe the important characteristics of the tool.

2.1 Graphical User Interface

MMIT presents a graphical interface to its users. From top to bottom, the MMIT interface contains a menu bar, a scrollable text window, and three scrollable lists and a message area (side by side) (Fig. 1). The menu bar is minimal and standard. The text window is used to display mission document pages.

The three lists and the message area display information that was not available to Milo's users. The first list shows the dictionaries (or document sets) which are available and cross-referenced for access by MMIT. The second list shows either a list of dictionary entries (if no particular entry has been selected and displayed),

or it shows the cross-referenced entries, of each dictionary, to the entry shown in the text window. The third list displays a history of the session by listing the names of all prior entries selected by the user. The message area is used to notify the user of errors or other system messages.

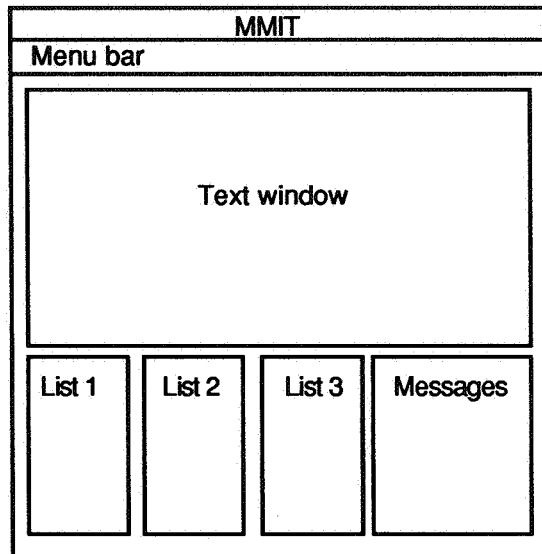


Figure 1. Layout of MMIT User Interface

2.2 Use of Standard Text Files

A primary design goal of MMIT was to permit the use of existing, unmodified document files. The MMIT prototype uses text files that were electronically transferred from the Milo system to a workstation. The only modifications to the files were simple removals of word-processor-specific formatting characters, and this process was automated using scripts.

2.3 Simple Cross-referencing

Because mission documentation is interrelated and its use is often centered on this relational nature, the provision of a user-friendly cross-referencing capability in MMIT is a primary feature of the tool. However, because we wanted to make as few modifications to the document files as

possible, the nominal hypertext approach, placing hypertext markers within the files, was not desired.

Instead we chose to construct indices into the documentation which MMIT would read and then display in its scrollable lists. This task was significant because the prototype included nearly 3000 document files. We were able to automate the task because the names of the index entries that were referenced in the documentation were the same as the filenames containing the information on that entry. We let the computer search through the documentation and build our cross-reference table for us.

2.4 Expandability

Because the document files are left unchanged, and because the cross-reference indices are built using automated scripts, MMIT is fully expandable to encompass new sets of related mission documentation. The only required modification would be to the MMIT initialization files which contain the cross-reference indices, and this is an automated process.

2.5 Object-oriented

MMIT uses object-oriented programming techniques. We believe that this practice results in software with higher reusability, modularity, maintainability, and reliability.

2.6 User Evaluation

User evaluation was an important part of the MMIT prototype development process. User comments helped to refine the user interface and correct problems with the basic capabilities. During the prototyping work, MMIT was evaluated by members of the Galileo MCT who guided the development such that they now use the prototype in their daily operations.

Since that time, MMIT has also been demonstrated to members of the Multimission Control Team and the Mars Observer Spacecraft Team. These interactions have enabled us to construct a list of capabilities for a new version of MMIT that will be even more useful and cost effective in operations.

3. PLANNED CAPABILITIES

3.1 Motif Compliance

The most visible change to the MMIT user interface is the planned modification to comply with the OSF/Motif™ window manager and style guidelines. Motif is the standard windowing style for JPL's telemetry system, the Multimission Ground Data System (MGDS). Since MMIT will interface directly with MGDS and have the same users as MGDS in the operational setting, the transition to Motif is a logical and significant step towards making the tool more usable.

3.2 Decommuration Map Generation

On board the spacecraft, data from the many sensors and instruments are assembled in a structured order before the telemetry stream is radiated back to Earth. This process is called commutation. Once the telemetry is received, the data stream is decommutated back into its original elements, called channels.

In MGDS, the process of telemetry decommutation is guided by a software decommutation map, or decom map, which specifies where each channel occurs in the telemetry stream and how many bits the channels contain. The decom maps are written in a high-level, MGDS-specific language called DMDL. The decom map files are long and complex since there are typically thousands of spacecraft channels and the process of decommutation varies with bit-rate, mode of spacecraft operation and mission phase. The process of editing decom maps is prone

to human error, and yet correct decom maps are critical to the interpretation of realtime telemetry.

A future capability of MMIT will allow users to generate and update decom maps directly from the telemetry channel documentation. With proper change control, we feel this can greatly improve the reliability and efficiency of the decom process.

3.2 CPT Generation

The MGDS system uses a Channel Parameter Table (CPT) to group telemetry channels according to their relevant spacecraft subsystems, to assign channel types, to set channel alarm conditions, to specify data number to engineering unit conversions, and invoke more involved channel processing. The CPT is created and updated from channel definition commands and is then converted to a binary form that is read by MGDS on start up (much faster than ASCII format).

It is not unusual for the CPT to change on a daily basis because channel alarm conditions constantly change over the course of a mission. The generation of a binary CPT is an involved one, so generally the MCT uses a binary CPT as a baseline with a supporting ASCII file of channel definition commands read in as changes from the baseline. This supporting file grows as new updates to the CPT are added. On a regular basis, the supporting file is integrated with the binary CPT and the procedure begins again.

This process could be entirely automated by generating the CPT directly from mission documentation. This type of automation has been used for the Deep Space Network (DSN) monitoring channels (Ref. 3). We plan that the next version of MMIT will be capable of such automation for all the spacecraft channels as well.

3.4 Anomaly Diagnosis Support

MMIT can also be used to support anomaly diagnosis, with relatively few modifications. By including existing anomaly reports in the supported MMIT mission document set, users would be able to cross-reference channel and command information with previously reported anomalies. In a realtime setting, this could speed initial diagnosis for recurring anomalies. For non-realtime anomaly isolation, such a reference tool could reveal patterns of anomalies and report previous anomaly resolutions in a quick and efficient manner. This capability would also allow projects to share ground system anomaly reports, and thereby increase the diagnostic reference capacity of the tool. Finally, it would be a significant step towards electronic anomaly reporting and processing.

3.5 Sequence Checking Support

Flight rules are operational constraints for spacecraft. MMIT already includes the flight rule documentation as part of its mission document set. Work is in progress at JPL on a syntax that specifies "flight rules" for command sequence checking programs (Ref. 4). Future versions of MMIT may be able to use this syntax to serve as an electronic reference for sequence checkers.

4. LIMITATIONS AND SOLUTIONS

From user evaluations and previous design decisions, we understand that MMIT has some limitations currently. As more capabilities are added to the tool, we can see other possible limitations rising to greet us. Here are our most significant concerns.

4.1 Indexing with Filenames

Currently, the mechanism for cross-referencing documents in MMIT is based on initialization files that are

read at application start up. These files are generated in an automated manner using scripts outside the MMIT program. As previously mentioned, this automation is possible because the document filenames are derived from the document entry name. As an example, the file containing the information on channel e-1004 would probably be named, e1004.tlm.

Although this is a legitimate design for MMIT because mission documentation has this structure, the applicability of MMIT to other groups of documentation where this structure did not exist would be limited.

4.2 Cross-reference Automation

The method of cross-reference automation used by MMIT requires a great deal of compute time to construct the cross-reference file. At run-time, however, this design allows the user to jump between related documents quickly by clicking with the mouse on the provided list of related documents. This results in a hypertext interface that allows linking of related documents without modifications to the documents themselves.

Although the design is an effective engineering solution, it does have limitations (Ref. 5). The hypertext links are based only on the internal references in the document. In other words, a user could be shown documents related to a single channel, but the user could not see a list of channels in the same subsystem as that single channel. Further, when the related document is displayed, the reference is not located or highlighted. However, the use of highlighting is planned.

In order to support new types of links, a cross-reference table would have to be generated for each type. With the large number of mission documents supported, run-time maintenance of the cross-reference table would result in unacceptable performance. This

limits the complexity of searches in MMIT to simple word searches and to the cross-reference tables already generated. Fortunately, because the type of documents supported and the typical uses for MMIT are known, cross-reference tables can be constructed in advance to support the large majority of queries.

4.3 Response Time

The large number of mission documents can result in unacceptable response times to users. So far, we have been able to compensate for this with design decisions such as the use of cross-reference tables over dynamic searches each time a document is displayed.

In the future, MMIT may become performance limited as new types of documentation (such as the anomaly reports) are added to the mission document set, or as the new capabilities of CPT and decom map generation are added. Almost certainly, CPT and decom map generation could become separate processes which are started by MMIT and which notify the user when complete. We feel the current relative growth in computing power will keep pace with (if not exceed) the relative increase in mission documentation.

4.4 Configuration Control

As MMIT becomes more integrated with critical operations tasks like CPT and decom map generation, the control of changes to the mission documentation becomes more important. To support critical operations requirements while allowing mission information to be distributed without fear of unauthorized changes, we propose to develop MMIT in two forms, an editor and a viewer. The MMIT Editor would only be supplied to cognizant individuals with the authority to change the documentation (so that the MCT could modify channel alarm limits for instance). The MMIT Viewer would allow a user to

reference and display all mission documentation, just like the Editor, but without the capability to alter the information viewed.

5. ACKNOWLEDGMENTS

This work was done at the Jet Propulsion Laboratory, California Institute of Technology, under a contract from the National Aeronautics and Space Administration. We would like to acknowledge the work of the technical staff in the OEL and the JPL Mission Operations Teams for their enthusiasm and support. We would like to especially thank John Louie for his exceptional technical support of the MMIT prototype development.

6. REFERENCES

1. Flight Projects Support Office. 1991. *Advanced Systems Office Technology Initiatives Annual Report*, JPL internal document D-9074, 25-30.
2. Murphy, S., Miller, K. and Louie, J. 1992. "Object Oriented Operations (O³) for Mission Control and Data Management", to be published, *Space Ops '92 Conf.*
3. Hurley, Daniel and Curran, Patrick. 1992. *Multimission Monitor Channel Dictionary*, JPL internal document D-9595.
4. Alkalaj, Leon. 1992. *Towards a Specification Language and Programming Environment for Concurrent Constraint Validation of Spacecraft Commands*, JPL internal document D-9921.
5. Glushko, Robert J. 1992. *Successful Hypertext Projects*, ACM Conference on Human Factors in Computing Systems, Chi '92, conference tutorial, 98-112.

AMFESYS: MODELLING AND DIAGNOSIS FUNCTIONS FOR OPERATIONS SUPPORT

N 94 - 23 887

J Wheadon

ESA/ESOC European Space Operations Centre
Darmstadt, Germany

ABSTRACT

Packetised telemetry, combined with low station coverage for close-earth satellites, may introduce new problems in presenting to the operator a clear picture of what the spacecraft is doing. A recent ESOC study has gone some way to show, by means of a practical demonstration, how the use of subsystem models combined with artificial intelligence techniques, within a real-time spacecraft control system (SCS), can help to overcome these problems. A spin-off from using these techniques can be an improvement in the reliability of the TM limit-checking function, as well as the telecommand verification function, of the (SCS). The paper describes the problem, how it was addressed in the study, including an overview of the "AMF Expert System" prototype, and proposes further work which needs to be done to prove the concept. The Automatic Mirror Furnace is part of the payload of the European Retrievable Carrier (EURECA) spacecraft, which was launched in July 1992.

1. INTRODUCTION

The recently-introduced technique, of transmitting spacecraft telemetry (TM) to the ground in the form of packets, introduces new problems of processing the data to provide the operations engineer with an accurate view of evolving spacecraft status.

The older TDM telemetry delivered regularly-sampled data in a fixed "TM format" structure, which itself gave a good approximation to successively sampled "snapshots" of spacecraft state; for most practical purposes, the operations engineer had only to display data on a format-by-format basis to understand what the spacecraft was doing.

Packet TM, in contrast, delivers data which are irregularly sampled, and data may even arrive in different packets out of chronological sequence (with respect to the timing of the events which they report). The principle is to deliver data from individual subsystems, only as fast as is needed to track how their status is changing. This means that the spacecraft designer can report a wider range of information for a given downlink capacity, compared to what TDM telemetry allows, but it also makes the processing on the ground more complex.

Most current ground-based Spacecraft Control Systems (SCS), which handle packetized TM, maintain a kind of "pseudo format" (PF) in computer memory by recording the latest values of TM data received, from whatever packets contained them. Although this means that changes to the PF occur irregularly, it gives an acceptable picture of the evolution of the spacecraft state over time, so long as the updated values in the TM arrive at the SCS in true chronological sequence. But when different packets report data out of sequence, or when "on-off" event report packets become lost in transmission, the PF will be inconsistent with the spacecraft.

Limit checking on TM parameters is usually performed by the SCS on the basis that the operating mode of each on-board subsystem can be derived simply from status information in the current TM format or PF, and that in most cases, a fixed set of limits can validly be associated with each such mode. This paradigm works well enough given a steady state of the spacecraft, although it often produces spurious limit alarms during sequences of state changes when switching from one operating mode to another. Spurious alarms will also occur if the PF adopts inconsistent states as described above.

Of course, spacecraft designers should try to make life easy for the operations engineer by ensuring that rapidly-changing data are reported often, that events such as mode switching are reported promptly, and that all information needed for spacecraft control is transmitted in true chronological sequence. ESA has defined standards which aim to promote such good (considerate ?) design, but historically, there has often been a tendency to make compromises in the spacecraft design which push problems of operation onto the ground segment. It should however be recognised, that increasing the complexity of the operations engineer's task in understanding what the spacecraft is doing, also increases the risk of not meeting mission goals; the operations engineers and technicians have to work reliably, day in and day out, over a period of years.

Mission operations also become more problematic for the ground segment when dealing with a spacecraft which does not continuously supply telemetry in near real-time, as for example a near-earth orbiter like EURECA (which itself uses packet TM). Typically these spacecraft have a ground station coverage (visibility from the station) of only about 10% of an orbit period of about 90 minutes. TM data are recorded on-board, and dumped at high speed over the ground station, in parallel with a "real-time" stream to give the operations engineer a "quick look" at the spacecraft status.

Such low-coverage science missions may have a high degree of on-board autonomy, performing complex measurement sequences. They are controlled during each orbit by an on-board Master Schedule of commands, previously uplinked from the ground. The operations engineer must largely rely on the autonomous functions, and can make only a few checks on the state of the spacecraft as it passes over the ground station, taking account of the last known state of the spacecraft and of the Master Schedule commands which should have been executed during the last orbit. The SCS maintains a display of the Master Schedule, and verifies execution of as many of the commands as possible, mainly by checking defined "verification parameters" in the real-time TM. It also performs mode-based limit-checking on the latest values in the TM format or PF, as described above.

Diagnosis of apparent spacecraft malfunctions or anomalies under these circumstances will be difficult to do in real-time during the station pass (about 7 minutes); spurious anomalies may be flagged by the SCS from time to time, and it will be essential not to take precipitate action when these occur. The operations engineer thus will have no choice but to rely heavily on the on-board safety mechanisms to act correctly when real failures occur.

2. APPLICABILITY OF MODELLING

The actions of the spacecraft operations engineer are in effect dependent on the his view of the state of the spacecraft at any moment, and on the need to act (eg command changes of mode), but respecting defined preconditions.

If the picture given by the PF ("pseudo format") contains inconsistencies, the operator will have to relate what he sees there to a mental image of what the spacecraft should be doing, and try to reconcile the two. He will also find himself having to predict the evolution of spacecraft status in the case of a low-coverage mission with high on-board autonomy, whether the telemetry is actually packetised or not, at times when the spacecraft is out of contact with the ground station.

In effect, the operator will have in mind a model of spacecraft, so why should we not help him by

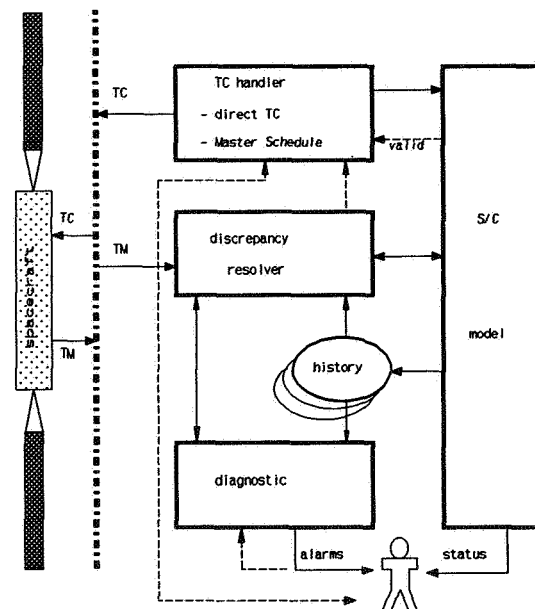


Fig 1: Proposed Concept

building such a time-driven model into the SCS ? The model can then become the reference for limit checking, as well as for command verification. For a low-coverage mission, the model can show the operator what the state of the spacecraft should be, at times when it is out of contact with the ground station. The model can include functions to accept and correctly emulate the spacecraft's handling of telecommands, both direct, and via a Master Schedule if the mission uses one. Fig 1 shows the proposed concept.

Classical "limit checking" in the SCS would then be replaced by a process of "discrepancy resolution", where discrepancies between the model state and the arriving TM data would be resolved by a knowledge-based reasoning process. Only if the TM data and the model were found really to be inconsistent, would out-of-limit alarms be raised to the operator, together with a possible diagnosis of the cause. Reconciliation of the model with the TM could be achieved by adjusting the model state according to the diagnosis, either automatically (in cases where the diagnosis is known to be reliable), or under operator control. In the rare case where a TM parameter is judged to have become unreliable, it would be permanently flagged as such in the discrepancy resolver.

To perform telecommand verification, the model would correctly predict the related TM changes, and any which did not occur would be picked up by the discrepancy resolver. The SCS should continue to keep a log of the execution of all telecommands.

The spacecraft history required for near-real-time evaluation and control would be represented by regularly recording the state of the model (with simple data compression if deemed necessary). Intermediate states at any moment of time could be obtained by extrapolating (modelling) forwards in time, but taking account of adjustments which were made from time to time by the discrepancy resolver.

All raw TM data useful for spacecraft control should also be filed, such that it can be used in long-term spacecraft performance evaluation, and for investigating and correcting possible deficiencies in the model. All payload data would continue to be processed and/or delivered to end-users as is done at present.

3. AMFESYS STUDY

A modestly-funded ESOC study contract completed in March 1992 has done some initial work to demonstrate the feasibility of the above approach to spacecraft control. A time-driven model of the Automatic Mirror Furnace (AMF) payload subsystem of the EURECA spacecraft has been developed, and integrated with a TC schedule handler, a TM tracker, a discrepancy detector and a diagnosis module. Fig 2 shows the architecture of this AMFESYS ("AMF Expert System") prototype, which is implemented on a Sun SPARC 2 workstation using the C-based object-oriented-programming tool ProKappa (Intellicorp).

The AMF is one of the most complex of the EURECA payloads, although much of the complexity is effectively hidden from the spacecraft controller because of the high degree of autonomy afforded by its embedded microcontroller. The scientific purpose of the AMF is to make experiments in the growth of crystal structures under micro-gravity conditions. The AMF oven comprises an ellipsoid mirror with a hole in the shell at each end of the major axis. One of 7 available halogen lamps is inserted through one hole, such that it radiates from that focus of the

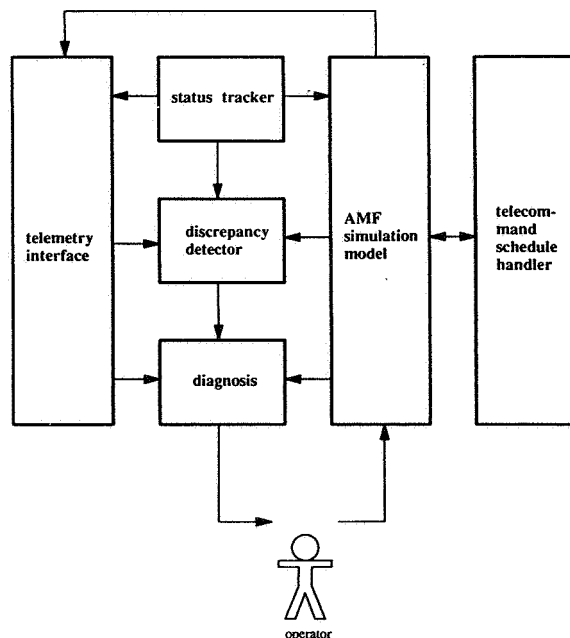


Fig 2: AMFESYS Architecture

ellipsoid onto the other focus. One of 24 cylindrical sample holders (ampoules), containing the crystalline material to be melted and re-crystallised, is inserted through the other hole, heated up to melt the crystals, and then withdrawn and simultaneously rotated about its axis at programmable rates through a cooled neck around the hole in the oven shell.

The 7 lamps are stored on a "lamp disk", whose axis is parallel to those of the lamps, and which transports them around and over the oven insertion hole. The lamps are driven in and out of the hole by a motor mechanism. The 24 samples, some of which are actually calibration probes, are stored on a similar disk at the other end of the oven, and are moved in and out of the oven by a motor mechanism which can also rotate them on their own axis.

Essentially, by monitoring temperatures and other information from along the sample, its holder, the cooling system and the oven shell, it is possible to gain an insight into the growth process (which is not simulated by AMFESYS !).

The AMFESYS model can simulate all operating cycles of the AMF as a function of time, and of the appropriate input telecommands which configure and program those cycles. The model simulates the status of all the mechanisms, and emulates the processing and measurement cycle control being done by the AMF microprocessor. The input of the required telecommands for any desired demonstration test case is effected by feeding to the AMFESYS model a stream of time-tagged TC history records which it interprets as a schedule of commands for execution at the stated times. This thus emulates the reception, by the real AMF, of commands from the EURECA On Board Data Handling System (OBDH) which processes the on-board TC Master Schedule. On the other "side", the AMFESYS TM tracker inputs a (history) stream of corresponding AMF housekeeping TM packets, using the packet time stamps to synchronise the handling of these TM packets with the currently-simulated time of the model. The model time for any test run is therefore set up to run through the time period covered by the corresponding TM and TC history streams.

The discrepancy detector module tries to deal with discrepancies between the TM parameter values, and the state of the model, either by ignoring them

if they are small enough (eg minor differences in measurements from analog sensors, or transient status inconsistencies), or by modifying the current status of the model if a reasonable and "nominal" explanation can be found to explain the differences. For example, the heating-up of the sample may sometimes take less time than the model predicts, thus resulting in an apparently premature start of the sample withdrawal, but this could be explained by the discrepancy detector, and it could advance the processing cycle in the model correspondingly.

If no "nominal" explanation for the discrepancies can be found, the diagnosis module is called on to make a more thorough analysis, which will usually result in the conclusion that a malfunction of some kind has occurred. The diagnoser presents its diagnosis to the operator, and proposes a corresponding "forced" change to the status or performance of the model, to correspond to the malfunction. The operator may accept the diagnosis and authorise the change to the model, or simply reject it, in which case AMFESYS will most likely throw it up again within a short time, or he may change the model himself because he decides that his diagnosis is the true one.

One could claim (although many would disagree) that the ability of the diagnoser to force changes to the model under the supervision of the operator, and in particular those changes which affect subsequent performance of the model (such as marking certain components as "failed"), constitutes a primitive form of adaptive "learning" about degradations of the real AMF, under the guidance of the operator. This is a little analogous to a pupil who says to his teacher (AMF engineer) "I think that motor1 has failed, don't you ?" to which the teacher will either agree, or supply his own better judgement.

Either AMFESYS will "learn" about degradations in the AMF, by recording them in its model under the operator's supervision, or the operator can impose the new knowledge directly. An "OK" status slot is defined for the general class *AMF devices* which is the parent of all AMF hardware object subclasses (eg switch, motor, spindle, gear, storage disk...). It is thus particularly easy to "fail" any object via the ProKappa Object Browser. However, it is only sensible to do this where there an object method which can use this information,

or a corresponding diagnostic rule to detect the failure.

AMFESYS can recognise and adapt its model to six non-trivial anomalies:

- 1: on-defective sample end-position switch
- 2: defective sample position sensor pot
- 3: spare lamp defective
- 4: sample code unreadable
- 5: wrongly-coded sample
- 6: lamp code unreadable

In the real AMF, all of these failures result in some sort of attempt by the AMF microcontroller to circumvent the problem, and an "exception packet" will eventually be emitted in the TM. AMFESYS can however already detect and correctly diagnose these anomalies from the housekeeping TM alone; in fact it does not use report or exception packets yet, although there are plans under consideration to make it do so.

4. CONCLUSIONS

The study was completed in March 1992, and the results demonstrate the feasibility of using a model as the basic reference for TM parameter limit checking, combined with automatic reasoning about the discrepancies, and adjustment of the model state under operator control. The available funding has restricted the study to demonstrate the technique for only one spacecraft subsystem, but nevertheless the AMF is a complex device, and entirely suitable as a "test case" for this type of experimentation.

As part of the task demonstrating the viability of the proposed approach to spacecraft control, an analysis was made of what monitoring facilities the user interface of such a system should give to the operator. The AMFESYS was provided with a user interface which not only shows him the evolving state of the model, but also a separate display showing the comparison between the incoming TM values, and the corresponding values derived from the model state. It is thought that the operator would normally use just the model status display; the "TM comparator display" would be useful when an anomaly occurs, especially during the early lifetime of the model when the users might be rather sceptical of the system's ability to handle all operating conditions. The techniques proposed here will need extensive validation in an

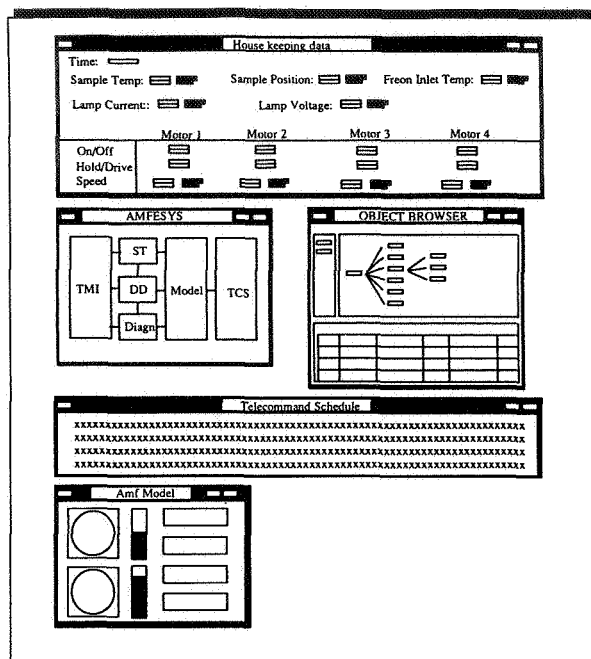


Fig 3: User Interface

environment which represents as closely as possible the real operational world.

The work of the study described above now needs to be extended further, to investigate the problems which may occur in extending the scope of the concept, eg to a range of spacecraft subsystems, in handling historical information, in coping with loss of TM packets, and in handling the non-chronological arrival of TM information. AMFESYS also seems to be a potentially good starting-point to develop the concept further by doing some experiments in model-based hypothetical reasoning, because of its architecture, and also since ProKappa allows the dynamic definition and instantiation of new rules and object classes.

Work needs to be done to set guidelines for the approach to the design and implementation of the models, as well as on the question of how to accommodate such models within the hardware and software infrastructure of future SCS. ESOC's experience over the last 20 years, in developing and using full spacecraft system simulators for ground system and procedures checkout and for operator training, provides a solid foundation for a methodology for the development of models integral with future SCS. However, it is important to keep separate the development of intra-SCS

models from the development of system simulators, which provide an essential independent reference against which to test the behaviour of the ground system. There are questions of both a technical and strategic nature to be considered.

The use of models in SCS is a technique which can be introduced gradually, first concentrating on spacecraft subsystems or functions whose behaviour is simple to model. For every additional subsystem whose TM housekeeping data can be monitored using the model paradigm, there will be a corresponding improvement in the operations engineer's understanding of what the spacecraft is doing. This will allow him to concentrate his attention on events that really matter, and so reduce the risk of human error in operations.

5. REFERENCES

1. "Model-based reasoning for operations support for the Automatic Mirror Furnace in orbit". R Dobiasch, G Schweizer, J Wheadon. Proceedings of the Workshop on AI and Knowledge-based Systems for Space, ESTEC, May 1991.
2. "A model-based approach to fault diagnosis and health maintenance in a spacecraft power subsystem". S Conway, AI Dept University of Edinburgh 1987.

6. ACKNOWLEDGEMENTS

AMFESYS was created as part of an ESA-funded study done by ITV, Karlsruhe (prime) and Siemens Austria, Vienna. Thanks are due to Prof G Schweizer, and G Theiss of Karlsruhe University, who analysed the problem domain, principally the functioning of the AMF. Thanks are also due to R Dobiasch of Siemens who performed the complete design and implementation of the software.

COMMAND AND CONTROL FOR THE ISRAELI LEO SPACE PROGRAM

Y. Barnett, E. Gal-Ezer, G.Herman, A. Klein, B. Walzer
Israeli Aircraft Industries (IAI)/MBT
P.O. Box 105
Yehud, Israel 56000

O. Cohen, Z. Lenefsky
Decision Systems Israel
Israel

*Paper presented but not submitted to proceedings
as of December 14, 1992.*

N94-23888

MARS OBSERVER SCREEN DISPLAY DESIGN FOR A MULTIMISSION ENVIRONMENT

Roy L. Chafin
Space Flight Operations Section 371
NASA Jet Propulsion Laboratory
Pasadena, California, USA

1. INTRODUCTION

The Multi Mission Control Team (MMCT) is responsible for support to real time operations of the Mars Observer Mission. The team has the responsibility for monitoring the ground data system for the integrity of the telemetry and command data links. It also supports the Mars Observers Spacecraft Team in monitoring spacecraft events. The Data Monitor and Display subsystem (DMD) workstation provides the data interface with the ground data system.

DMD workstation displays were developed to support the Mission Controllers in accomplishing their assigned tasks for supporting the Mars Observer mission. This paper presents the display design concepts that were used in the Mars Observer MMCT displays to minimize the cognitive demands on the controllers and enhance the MMCT operations.

1.1 Data Monitor and Display Subsystem

The Data Monitor and Display subsystem (DMD) is the controllers window into the spacecraft and the ground data system. The DMD is a workstation that provides a variety of formatted data displays to the controller. The displays present both spacecraft telemetry data and ground system monitor data. Some displays are preplanned and developed prior to the operations in which they are used. These are called fixed displays and are quite versatile in format and content. Other displays and plots can be created in real time. These displays have limited formats but flexibility in content. These are called list or message displays. They can be rapidly generated by the controller as needed.

The MMCT display repertoire provides a mix of displays appropriate to the needs of the MMCT controllers. Figure 1 illustrates both the fixed and message display formats.

2. DISPLAY DESIGN CONCEPTS

2.1 Operational System Context

Displays should be developed in context of the operational system. That is, displays should be developed that respond to the operations systems requirement (Ref. 1).

In order to be effective, displays must focus on the need of operations system. The display should focus on specific information that is required to complete a task. Superfluous display information creates confusion and requires extra effort from the controller for filtering. Of course, missing display information reduces the display effectiveness.

2.2 Knowledge based vs Procedural based

One of the basic parameters of designing an operations system is whether the operation will be Knowledge based or Procedural based. That is, will the normal operation be based on the operators knowledge of the process or will the operator normally be guided by preplanned procedures. The advantage of Procedural based operations design is that the skill requirements on the operator is less than for a Knowledge based operations system. Thus the operator cost can be less for Procedural based system than for Knowledge based system. Procedural based system design can be used when the basic process is well known and relatively simple (i.e. procedures can be written), and the basic system

is stable (i.e. procedures are continuously valid).

When the basic system process is not well understood or the process changes frequently, adequate procedures are difficult, therefore the system must be operated as a Knowledge based system. This requires that the operator be sufficiently knowledgeable of the system process so that he can recognize when problems occur and can formulate plans to resolve the problems. The advantage of a Knowledge based system is that more complex and variable precesses can be handled. Preplanning is minimized. Another advantage is that the operator responds to problem when they occur, that is when they are the most well defined.

2.3 Focus and Structure

To support Procedural based operations displays should be designed to ease the cognitive load on the controller. The cognitive load on the controller can be eased by focusing the display on only the data needed for the tasks that he is performing. Thus he does not need to filter data. Figure 1 illustrates both focused and unfocused displays. Relationships and conditions can be emphasized, such as color coding the display elements that are active as against those that are not active. Visual and graphic display elements can be used to indicate the condition of more abstract conditions.

2.4 Hierarchic Displays

System troubleshooting is enhanced by providing a hierarchic system of displays. They start with a high level system displays containing the basic elements of the system, preferably in a block diagram format, with GO/NOGO indications. Lower level displays expand the displayed information for the system elements identified as faulty by the higher level display. This display concept can be continued until the lowest level of interest is displayed.

2.5 Compatibility

Compatibility eases the controllers cognitive load by presenting like things the same on different

displays. The cognitive effort is reduced because the display elements are recognized and do not have to be deduced (Refs. 2 & 3).

Compatibility also refers to naming the display elements with names that are the same as those used in other activities i.e. common names. Conventions used in various displays should be similar. For example, in both the Down Link display (Fig. 2) and the Up Link display (Fig. 3) the existence of the telecommunications link is indicated by an arrow connecting the spacecraft box with the ground system box.

2.6 Example

The Down Link Display (Fig. 2) illustrates these concepts. The display is focused on the down link telemetry system. The spacecraft elements of the downlink are displayed in a box. The ground station elements of the downlink are displayed in a separate box. The display identifies the systems elements that are active by being presented in bold. The overall status of the down link is visually presented by the arrow from the spacecraft box to the ground station box. The existence of the arrow very rapidly identifies that the down link exists. The absence of the arrow indicates that the down link is broken and that no data is flowing.

3. TYPES OF DISPLAYS

Three types of displays were developed for the Mars Observer mission support. They are Multimission system displays, mission sequence displays, and controller discretionary displays.

3.1 Multimission System Displays

System displays provide the controller with the information about the overall performance of the system. They quickly indicate that the system is operating as expected or conversely, that something is wrong. The Ground Data System display (Fig. 4) illustrates this type of display.

System displays for mission controller (MMCT) need to support the controller in three areas,

configuration, performance, and sequence.

3.1.1 Configuration

The mission controller must confirm that the configuration of the ground data system and the spacecraft are proper for the transmission of telemetry data and spacecraft commands. That the planned system elements are up and operating correctly, and that the system elements are set with the proper parameters such as bit rate. The system displays should help the controllers verify the integrity of the spacecraft/ground data system.

3.1.2 Performance

The mission controller display should also provide an indication of the performance of the data and command systems. That is, the signal levels and signal to noise ratios (SNR) are as predicted. If the performance is not as predicted, action is required to investigate and resolve discrepancies before data degradation occurs. The ground data system display provides this information (Fig. 4).

3.1.3 Sequence

Occurrence of expected events are confirmed, and the occurrence of unexpected events is detected. For example, receiver lock up times are displayed in the ground data system display to confirm that the tracking station is operating correctly and that the data is flowing (Fig. 4).

3.1.4 Users Responsibilities

The mission system displays allow the Mission Controller to meet their responsibilities to the project in confirming the integrity of the data and command systems.

3.1.5 Procedural Based

Top level system displays indicate the overall performance of the ground data system. They tend to support Procedural based operation in that they provide information to confirm preplanned configurations and performance. Lower level subsystem displays provide more detail about the configuration and performance and

tend to support Knowledge based operation. The high level displays are usually fixed DMD displays. The lower level displays may be fixed displays but often are list or message displays.

3.2 Mission Sequence Displays

These are displays that support the mission controller in confirming the sequence of spacecraft and ground data systems events. The sequence of spacecraft and ground data system events is preplanned and documented in the Project SOE (Sequence Of Events). The mission sequence displays provides the controller with information which allows him to confirm that the spacecraft and the ground system in fact operating according to the SOE. During some parts of the mission, the Mars Observer Spacecraft Team is staffed for only 40 hours per week. The MMCT controllers being on duty continuously, support the Spacecraft Team by monitoring the spacecraft.

The Spacecraft Team provides the standards and limits for the DMD displays. The MMCT responsibility then is to monitor the display for out of limit conditions. This removes the requirement that the controllers be trained in the spacecraft system. There are two basic kinds of mission sequence displays: displays for unique sequences and displays for repeated sequences.

Unique sequences are sequences of events that occur only once during the mission. Displays for unique sequences focus on the specific events as they occur and reduce the work load on the controller in that he doesn't have to interpret the SOE and search the telemetry data stream displays for the information that confirms the event. Since they occur only once, and for cost considerations, they may not be developed as fixed displays but as list or message displays.

Some spacecraft sequences are repeated many times, such as the spacecraft data recorder operations. These displays are valuable and should be well designed and well focused since they are used repeatedly. The DTR display (Fig. 1)

illustrates a repeated sequence display. They allow the controller to focus on just the data that pertains to the event occurring. The controllers become very familiar with the display and thus reduce their workload.

The mission sequence displays tend to be procedural based in that they provide the mission controller with information that allows them to confirm preplanned events. Fixed displays are cost effective for repeated sequence displays. List or message displays may be more appropriate for unique sequence displays.

3.3 Controller Discretionary Displays

The displays discussed above, must be designed in advance of their use. In addition to the fixed displays presented above, the DMD provides the flexibility for mission controllers to construct displays list and message displays in real-time. This is needed for unanticipated events.

Invariably, things happen that are not anticipated. Controller discretionary displays allow the mission controller to look at these events and evaluate their consequences to the mission.

Some times it is desirable to investigate events deeper in detail to better understand what is happening. This may be to confirm unanticipated spacecraft or ground data system operation. It may also be used by the mission controller for his own satisfaction so that he can understand the spacecraft or ground data system better.

The controllers knowledge of the spacecraft or ground system can often be used to anticipate problems. Controller discretionary displays allow the controller to investigate things that look suspicious and to avoid problems before they occur.

3.3.1 Knowledge based

The controller discretionary displays tend to be Knowledge based in that they are used to provide unstructured information that is used by the mission controller to develop his own understanding of the process. This

understanding allows the controller to take actions on situations that were not covered by procedures.

4. CONCLUSIONS

Well designed displays are developed by careful considerations of basic display concepts. They are well focused, in context, and compatible. These displays enhance the controller mission support by reducing his cognitive load.

Well designed displays are cost effective because they are easy to use.

5. REFERENCES

1. Chafin, Roy L. 1992. An Approach to the Design of Operations systems, Second International Symposium on Ground Data Systems for Space Mission Operations.
2. Nielsen, Jakob. 1989. Coordinating User Interfaces for Consistency, Academic Press, Inc. Boston, MA.
3. Shneiderman, Ben. 1987. Designing the User Interface, Addison-Wesley Publishing Company. Reading, MA.

U-92/142-2415	CE-92/263-08145	RC-04/2192-11	23:12.3
U-92/112-1412	ELK-401359501	(17EC428D)VIDEO UPDATE	2. S
C-0016 \$ DIRL F	N	ON	14:29:10
C-0051 \$ DRIHQD	15	STAND BY	14:29:07
C-0017 \$ DIRL TAP	DIR 01	FORWARD	14:29:12
C-0015 \$ DIRL HTP	STP 01	STOP	14:29:02
C-0013 \$ DIRL BO	TAPE 00	NOT BOT	14:29:11
C-0014 \$ DIRL TO	SLCT 00	TU, I	14:29:10
C-0018 \$ DIRL	TRK 00	END 31	14:29:07
C-0066 \$ DIRL	TRK 04	TRACK 1	14:29:11
C-0101 U DIRL	TAPE SPD	ON	14:29:12
C-0012 \$ DIRL	P1-HTR 00	OFF	14:29:11
C-0011 \$ DIRL	B-U/HTR 01		14:29:06
C-0501 U DIRL	CURRENT	21	14:29:11
C-0903 U DIRL	PC-TEPH	145	14:29:05
C-0317 U DIRL	-5W, FHR	215	14:29:00
C-0319 U DIRL	52U-REG	215	14:29:10
C-0262 U DIRL	SERV-ER	215	14:29:10

— **THE**

```

CPU=22:10:34
MTISB0 B#0
P8=1 #=000

```

DTR STATUS

*U=32/142-21:55:28 SCE=92/263-08148:01 CR-04/21/92-11:29:12
 *U=92/112-14:29:12 SCLX=401395801 (17EC4F2D)V/VIDEO UPDATE 2 5
 14:29:10 OFF 14:29:10
 14:29:10 STAND BY 14:29:10
 14:29:10 REVERSE 14:29:10
 14:29:10 RUNNING 14:29:10
 14:29:12 BOT 14:29:12
 14:29:12 EOT 14:29:12
 14:29:12 TU_2 14:29:12
 14:29:12 END 14:29:12
 14:29:12 TRACK_1 14:29:12
 14:29:12 TRACK_0 14:29:12

25 14:29:109
 143 14:28:58
 14:29:01 0.67V DEC LT
 14:29:01 0.67V DEC LT
 14:29:01 2.032 VPS
 16 2.032 VPS

72.5 MAHP
 20.67V DEC LT
 0.67V DEC LT
 2.032 VPS

C-0502 U DIR2_CURRENT
 C-0904 U DIR2_AC_TEMP
 C-0320 U DIR2_15V_P4R
 C-0322 U DIR2_12V_REC
 C-0202 U DIR2_SERV_ER

Operation	1	2	3
Recorder Power at SCET	ON 00:00:00	ON 00:00:00	OFF 00:00:00
Tape to Standby at SCET	RUNNING 00:00:00	RUNNING 00:00:00	STOP 00:00:00
Recorder Mode at SCET	REC_16K 00:00:00	PLBK_42K 00:00:00	STAND_BY 00:00:00
Tape Position	Forwd 4	Rev 5	BEGIN 1
Tape Direction	B 00:00:00	B 00:00:00	B
and Track Number	F 00:00:00	R 00:00:00	R
	E 00:00:00	E 00:00:00	E

DTR-TU

1	U=92/112-21185:28	SCE=32/233-0814H 01	BC=04/21/92-1129:23	S
2	1129:23/12	SL=44/01359501	1175C=28D/VINDE	UPDATE
3	C-0000	5 DIR3_PIR_ON	00	11:29:09
4	C-0001	5 DIR3_CHD	ON	11:29:09
5	C-0023	5 DIR3_HTR	STOP	11:29:10
6	C-0025	5 DIR3_HTR	STP 01	11:29:11
7	C-0025	5 DIR3_BO	TAPE 01	11:29:12
8	C-0026	5 DIR3_EO	TAPE 01	11:29:12
9	C-0030	5 DIR3_TU	SLECT 01	11:29:09
10	C-0095	5 DIR3_CLK	04	11:29:11
11	C-1016	5 DIR3_TRK	00	11:29:11
12	C-0103	5 DIR3_TAPE_SPD	00	11:29:11
13	C-0503	U DIR3_CURRENT	20	11:29:11
14	C-0505	U DIR3_RC TEHP	132	11:28:59
15	C-0323	U DIR3_6V_P4R	195	11:29:06
16	C-0325	U DIR3_12V_REG	216	11:29:00
17	C-0203	U DIR3_SERV_ER	1	11:25:09

Digital Tape Recorder (DTR) Displays

Fig. 1

```

CPU=19:22:11      MOTISB0 B#20700  Pg= 1 #-000
ERT=19:21:57      E#4627    H#3413  Q#3    DSS 45

DOWNLINK STATUS  S/C  MOT1  MOT2  InetLink  V
-----
| TRNC  OFF  ON  INHIBIT  |
| Coh/NonCoh INHIBIT  |
| Telemetry ON_  OFF_  |
| Ranging ON_  OFF_  |
| DOR OFF_  OFF_  |
| USD ENABLE INHIBIT  |
| Antenna LO_GAIN  |
|-----|
| Telemetry  |
| S/C MOT1 is con to S&E_1A  |
| MOT2 is con to TOS  |
| DSS Ch1: 4000 kps  |
| Ch2: bps  |
|-----|
| Antenna 148  |
| SNT 31  |
| Rcv Lock: INLOCK  |
| @ ERT 15:02:59  |
| AGC -142.4  |
| DSA Ch1:INLOCK Ch2:  |
| @ ERT 15:02:59 00:00:00  |
| SNR Ch1: 1384 Ch2:  |
| DOP way Samp Intv 10.0  |
| RNG Last Comp  |
|-----|

```

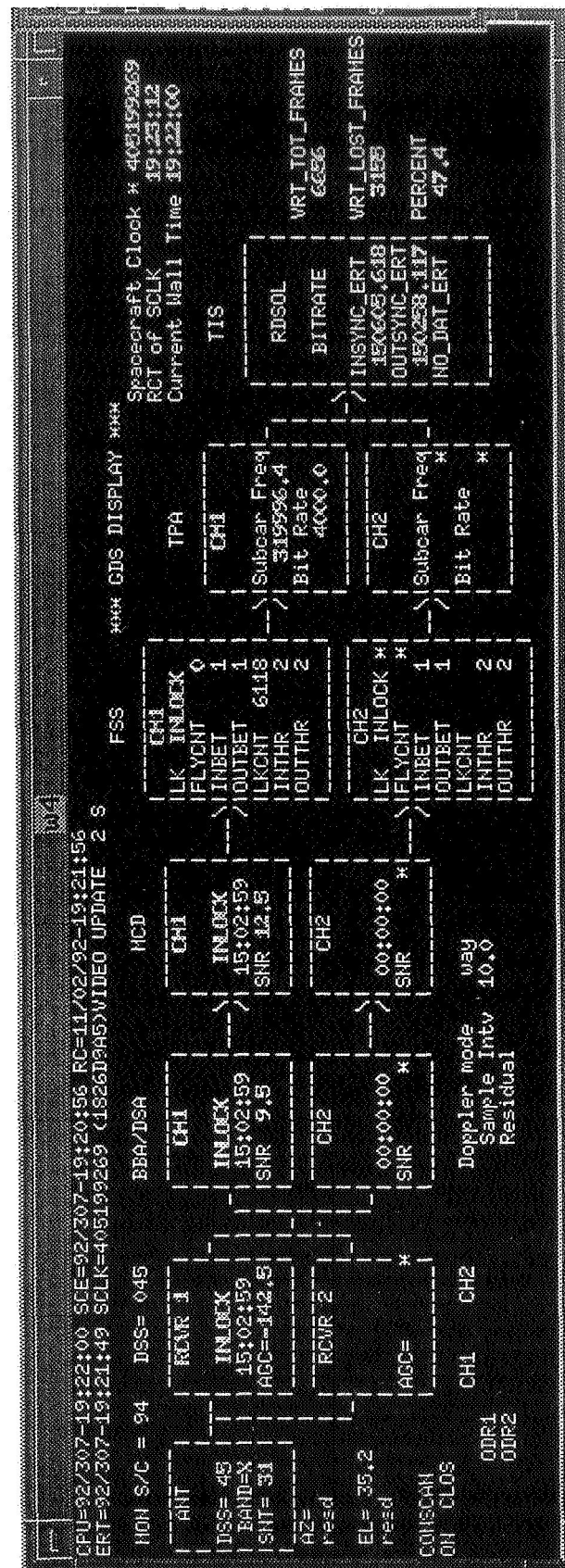
Down Link Display

Fig. 2

CPU=23:55:18		MOTISB0 B#0		Pg= 1 #=000	
ERT=00:00:00		M#0		Q#0	
E#0					
UPLINK STATUS					
S/C		HI_GAIN			
Antenna		MOT1		MOT2	
Rcv Lock		INLOCK		OOL	
at SCET		00:00:00		00:00:00	
Rcv AGC		-138.966			
CDU LOCK		INLOCK		OOL	
at SCET		00:00:00		00:00:00	
CDU SNR		20			
DSS		CDU1		CDU2	
CMD Bit Rate 62.5		62.5			
Up Link					
Exc Freq 42646573.1					
TX ID 15					
CMD MOD ON					
TX BEAM ON					
TX PWR 18					
TX ANT ID 15					

Up Link Display

Fig. 3



Ground Data System Display

Fig. 4

THE FOT TOOL KIT CONCEPT

N94-23889

Michael Fatig
Bendix Field Engineering Corporation
Seabrook, Maryland, U.S.A.

ABSTRACT

Flight operations and the preparation for it has become increasingly complex as mission complexities increase. Further, the mission model dictates that a significant increase in flight operations activities is upon us. Finally, there is a need for process improvement and economy in the operations arena. It is therefore time that we recognize flight operations as a complex process requiring a defined, structured, and life cycle approach vitally linked to space segment, ground segment, and science operations processes. With this recognition we have developed an FOT Tool Kit consisting of six major components designed to provide tools to guide flight operations activities throughout the mission life cycle. This paper addresses the major components of the FOT Tool Kit and the concepts behind the flight operations life cycle process as developed at NASA's GSFC for GSFC-based missions.

The Tool Kit is therefore intended to increase productivity, quality, cost, and schedule performance of the flight operations tasks through 1) the use of documented, structured methodologies; 2) knowledge of past lessons learned and upcoming new technology; and 3) through reuse and sharing of key products and special application programs made possible through the development of standardized key products and special program directories.

1. BACKGROUND

GSFC is responsible for low earth orbiting scientific satellites. The evolution of GSFC missions over the years has resulted in dramatic changes in operations. Increased mission complexity, increased coupling between space, ground, science, and operations, new technology in space and ground systems, larger spacecrafts and payload sets, multi-mission sets with coordinated science campaigns, heavily utilized and loaded shared resource support systems, and other changes over the last ten

years has changed the demands on the flight operations teams both in the premission phase and during orbital operations. For example:

1. Spacecraft ten years ago had several hundred to a thousand telemetry parameters, and only several hundred largely pulse type commands. Today the missions have 2000 to 3000 telemetry parameters, over 1000 commands, commands with many attributes, options, and subfields, and more complex processes for interaction between subsystems.
2. With the advent of realtime science (as we move from survey missions to more dynamic realtime studies) the nature of mission planning is changing, moving away from advanced planning to near realtime and realtime resource and constraint management.
3. The increased utilization of flight software has added onboard processes that compliment the ground process, often simplifying operations under normal conditions but complicating the fault detection, identification, and resolution process.
4. The trend back towards smaller missions which are largely ground station-only supported require handling these increased mission complexities in short duration realtime events (typically 8 to 12 minutes).

It is clear that the nature of flight operations has changed.

Finally, the GSFC mission development model is growing. We have more new missions than we have experienced, flight operations development personnel to handle them. We can no longer count on an experienced workforce, and must develop tools to guide new operations engineers.

Due to these challenges on operations, we must now recognize the complex and full life cycle operations process. Accordingly, GSFC has

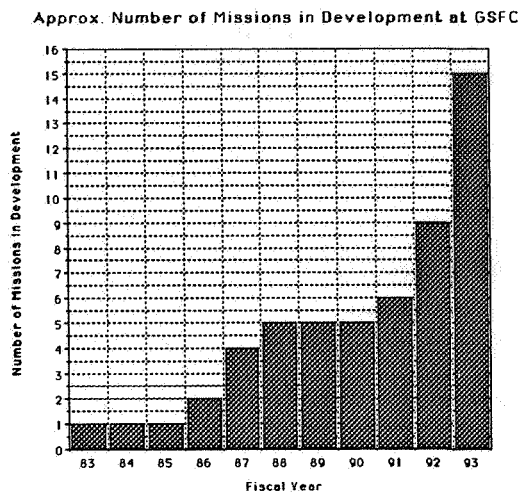


Figure 1: Mission Development Model

developed and implemented an operations engineering process. The intent is to build upon the principles of concurrent engineering, involving the user in the development from day 1, and building operations into the design. A cornerstone of this process is the FOT Tool Kit.

2. FOT TOOL KIT COMPONENTS

The FOT Tool Kit is the first comprehensive effort at collecting information to guide the operations development and implementation process. It has a full life cycle scope, is both forward and backward looking, and promotes reuse and sharing among Flight Operations Teams. It will assure a higher quality operations service built on proven principles of previous missions, and as a documented process allows for continuous process improvements. It consists of two main elements: 1) Methodologies, and 2) Support Tools.

2.1 Methodologies

Four methodologies are provided in the tool kit: 1) The Operations Engineering Methodology (OEM) document; 2) The Continuing Operations Methodology (COM) document; 3) The Flight Operations Management Methodologies (FOMM) document, and 4) The Training Methodologies (TM) document.

The Operations Engineering Methodologies document provides a work breakdown structure (WBS) for the operations effort required in the mission preparation phase. The WBS follows the operations engineering process. For each major WBS element, the document provides a template of information including a checklist of considerations, definition of key products and reviews, a process chart, and key inputs and interdependencies. All of the information is based on successful practices of past missions. It includes operations engineering processes for both within and across mission engineering, thereby supporting the mission engineering concept.

The Continuing Operations Methodology is a similar product based on a WBS for operations activities in the orbital operations phase. It provides recommendations for handling each task based on successful practices of previous missions.

The Flight Operations Management Methodology document, also based on a WBS, provides mission assessment models for defining the flight operations efforts, guidelines for schedule development, guidelines for management of the operations effort in both the premission and orbital operations phase, and methods for interacting with the space, ground, and science components of a mission (the mission engineer-

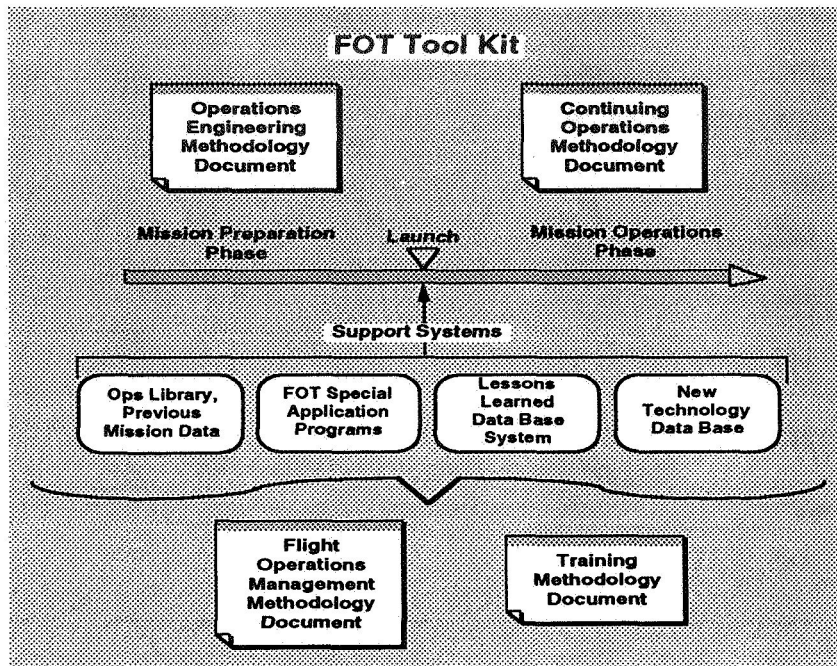


Figure 3: FOT Tool Kit

process improvement. The operations engineering process and FOT Tool Kit efforts are attempts at changing that. The challenge is to first collect the expert knowledge in a defined process, then employ continuous improvement principles in refining the process.

4. FUTURE ENHANCEMENTS

In its present state of development the FOT Tool Kit is a rudimentary collection of information, largely in written form. Enhancements to the Tool Kit would greatly increase its utilization and power. There are plans for development of a FOT life cycle assessment and cost model, a flight operations information collection and management system, a requirements generation

system, and key product development tools.

5. IMPLEMENTATION

The concept of a defined, recognized, and applied operations engineering process and the FOT Tool Kit which supports it has been gaining momentum and recognition throughout the GSFC community. Nearly all GSFC new missions (XTE, TRMM, SOHO, ACE, FUSE) are implementing operations engineering. This is a start, largely from a grassroots effort. There needs to be an investment in full scale development and implementation, as well as organizational changes to fully implement and therefore reap the benefits from it. In the meantime the grassroots effort will continue.

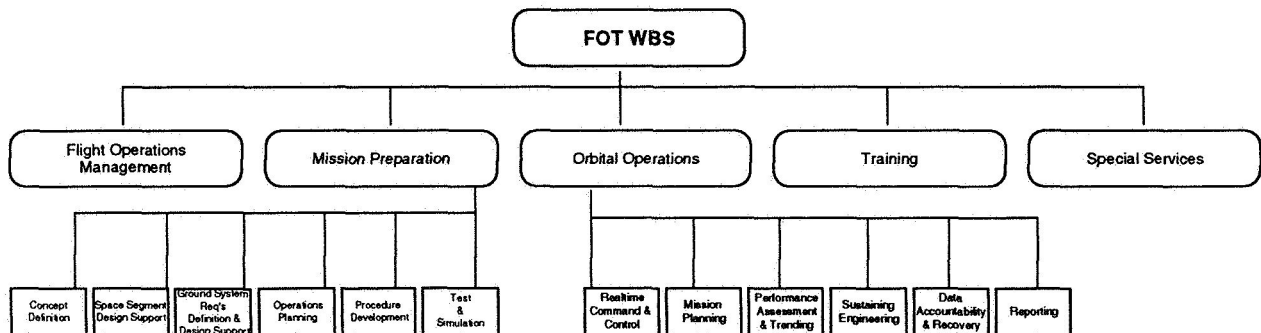


Figure 4: WBS Chart

GSMS AND SPACE VIEWS: ADVANCED SPACECRAFT MONITORING TOOLS

Douglas Carlton and David Vaules, Jr.

N94-23890

Computer Sciences Corporation
1100 West Street Laurel, MD 20707

Daniel Mandl

NASA/Goddard Space Flight Center
Code 511.2
Greenbelt, MD 20771

ABSTRACT

The Graphical Spacecraft Monitoring System (GSMS) processes and translates real-time telemetry data from the Gamma Ray Observatory (GRO) spacecraft into high resolution 2-D and 3-D color displays showing the spacecraft's position relative to the Sun, Earth, Moon, and stars, its predicted orbit path, its attitude, instrument field of views, and other items of interest to the GRO Flight Operations Team (FOT). The GSMS development project is described and the approach being undertaken for implementing Space Views, the next version of GSMS, is presented. Space Views is an object-oriented graphical spacecraft monitoring system that will become a standard component of Goddard Space Flight Center's Transportable Payload Operations Control Center (TPOCC).

Key Words: Spacecraft monitoring, graphical, visualization, mission control

of the spacecraft from the information on their displays. To assist the Gamma Ray Observatory (GRO) FOT with this visualization, the Graphical Spacecraft Monitoring System (GSMS) was developed. The GRO GSMS processes and translates real-time telemetry data into high-resolution 2-D and 3-D color displays that show the spacecraft's position and attitude relative to the Sun, Earth, Moon, and stars; its predicted orbit path; its instrument fields of view (FOVs); and other items of interest to the FOT.

As GRO GSMS has been found to be of high value to the FOT over the past 18 months, the next evolution of GSMS is now under development. Space Views will encompass all that GSMS is today, but will address making a generic graphical spacecraft monitoring system that can be easily tailored to support new spacecraft missions. An object-oriented approach is being used to achieve this goal. Space Views will also operate under X Window and Motif, in a client-server environment, providing network distributed displays across a mix of Unix-based workstations.

1. INTRODUCTION

Traditionally, the flight operations team (FOT), which is responsible for monitoring and maintaining the health and safety of a spacecraft, has performed its duties using alphanumeric text displays of telemetry data as its primary source of status information, augmented with graphical strip charts of a few select telemetry items. The operators must constantly interpret the data on their displays, building a mental picture of the spacecraft's condition. When a problem occurs, they must quickly determine the appropriate corrective action. Their ability to properly diagnose a problem depends, in part, on their ability to visualize the state

2. BACKGROUND

The Control Center Systems Branch (CCSB) at the Goddard Space Flight Center (GSFC) in Greenbelt, Maryland, is responsible for developing spacecraft control centers at GSFC. Current spacecraft missions supported by CCSB at GSFC include the International Cometary Explorer (ICE), Earth Radiation Budget Satellite (ERBS), Cosmic Background Explorer (COBE), Upper Atmosphere Research Satellite (UARS), Extreme Ultraviolet Explorer (EUVE), Solar Anomalous and Magnetospheric Particle Explorer (SAMPEX), and GRO. Each spacecraft mission has its own Payload

Operations Control Center (POCC), consisting of a Mission Operations Room (MOR) connected to, or containing, computer equipment hosting mission support systems.

From the MOR, the FOT provides round-the-clock support to its spacecraft. At scheduled times, the FOT makes a communication contact with the spacecraft that typically lasts about 20 minutes. During this contact, the FOT monitors telemetry data from the spacecraft on alphanumeric text displays that are color coded to show telemetry items with nominal values in green, marginal values in yellow, and out-of-tolerance values in red. The FOT relies on these color codes to help highlight potential problems with the spacecraft. The GRO MOR contains 12 display screens which must be monitored by 2 operators during each support period.

To further automate the POCCs and eliminate tedious, manual procedures for determining and monitoring spacecraft status, the CCSB directed CSC to develop the conceptual design for a graphical spacecraft monitoring system (GSMS). After concept approval, the CCSB tasked CSC under NASA's Research Technology, Objectives and Plans program to develop requirements and implement the system as a pilot project.

3. APPROACH AND REQUIREMENTS

The task of defining requirements for the GSMS was begun by creating a simple prototype, developing a few displays illustrating the capabilities of state-of-the-art computer graphics workstations. One display was a 3-D model of the GRO spacecraft that moved along its roll, pitch, and yaw axes in response to simulated data from a file. A second display simulated a video strip chart plotting several parameters at once in real time against separate left and right vertical scales, with a user-selectable time scale. A third display illustrated the GRO propulsion subsystem as an animated schematic diagram, showing the effects of a stuck valve by highlighting failed components in red, overlaid with alert messages. This simple prototype performed rudimentary display window management, allowing any display to be shown either full-screen or in a quarter-screen mode along with three other displays. The prototype also included basic popup menus for selecting display options with the mouse.

With the prototype completed, demonstrations were conducted for personnel from both current and upcoming spacecraft missions. Senior analysts from the GRO mission saw the potential benefits of the GSMS immediately and requested that it be developed into an operational system to support their FOT.

Working with GRO senior analysts, requirements for 11 basic displays were defined:

- Mercator map showing the orbit path of GRO, communication coverage of the Tracking and Data Relay Satellites (TDRSs), day/night terminator, and South Atlantic Anomaly (SAA) region of magnetic interference
- View of TDRS coverage of the Earth, as seen from the altitude of each TDRS
- View of GRO from its altitude, in its current position and attitude, shaded according to its orientation to the Sun
- View from the -X axis of the GRO spacecraft, with selectable FOV angles, showing reference frames for the bright object sensors (BOSs), fixed-head star trackers (FHSTs), and omnidirectional antennas
- View from the +X axis of the GRO spacecraft, with selectable FOV angles, showing reference frames for the fine Sun sensors (FSSs) and omnidirectional antennas
- BOS-1 and -2 FOVs showing shutter trip lines for closure due to the presence of the Sun, Earth, or Moon
- FHST-1 and -2 FOVs showing the position of the current selected guide star within the reference frame
- FSS-1 and -2 FOVs showing the position of the Sun within the reference frame
- Omnidirectional antenna-A and -B FOVs, with selectable FOV angles, showing reference frames for the BOSs, FHSTs, and FSSs
- High gain antenna (HGA) FOV showing the current position of a TDRS within the HGA communication coverage area

- Also defined were displays that were combinations of the 11 basic displays, designed to maximize the use of available screen space to efficiently communicate spacecraft information to the operator. In total, 25 displays were defined for selection by the operator.

Development of GSMS for GRO was bound by the constraint that this system, as a pilot project, not impact the GRO POCC ground support system. New requirements could not be levied on the GRO POCC to facilitate integration of GSMS with the POCC nor change existing POCC interfaces to other supporting systems such as the Flight Dynamics Facility (FDF). This meant that accessing real-time telemetry in the GRO MOR had to be accomplished via a terminal interface to the GRO applications processor (AP), which received and processed telemetry for display in the MOR. GSMS also required access to predicted orbital data from the FDF. Because no direct electronic link existed between the GRO POCC and the FDF, the only way to access this data was by having the necessary files written onto 9-track tape and hand-carried to the GRO MOR. With these constraints in mind, a system architecture for GSMS was developed that, in effect, made GSMS a "read only" application within the MOR. Figure 1 shows the overall system configuration for GSMS.

```

graph TD
    AP[APPLICATIONS PROCESSOR] --> TD[TELEMETRY DISPLAYS]
    TD --- PC386[PC-386]
    PC386 -- "EXTRACTED TELEMETRY DATA" --> SGI[SGI IRIS 4D/20]
    9TT((9-TRACK TAPE)) -- "FLIGHT DYNAMICS PREDICTED ORBITAL DATA" --> SGI
  
```

Figure 1. GRO GSMS System Configuration

The GSMS software on the SGI workstation is the core of GSMS, presenting the 25 defined 2-D and 3-D displays on a high-resolution color monitor and handling the user interface for selection of the displays to be shown, selection of the display options, and placement of the display when in the quarter-screen viewing mode.

The GSMS user interface features immediate visual feedback to the operator. When the operator invokes a popup menu of display options by pressing a mouse

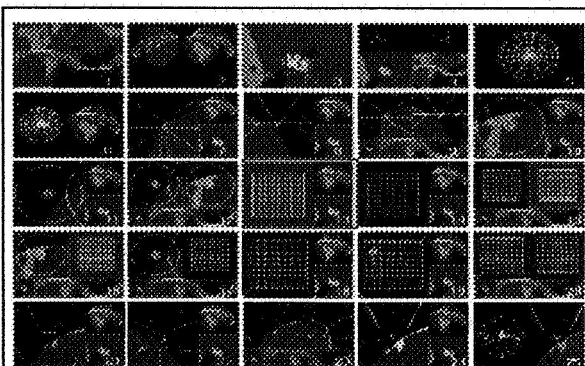


Figure 2. Iconic Display Selection Menu

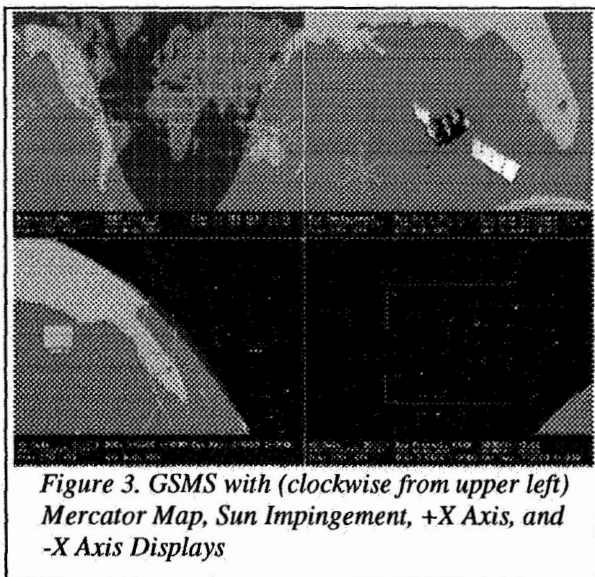


Figure 3. GSMS with (clockwise from upper left) Mercator Map, Sun Impingement, +X Axis, and -X Axis Displays

button, the effect of any selected option is immediately shown in the display, which continues to update even while the option menu is present. This allows the operator to see the effect of the option immediately and decide whether to accept it while the menu is still available for making choices and selections. Another feature of the GSMS user interface is its window management capability, which allows four displays to be viewed simultaneously in a quarter-screen mode, as shown in Figure 3, or a single display to be viewed full-screen, as shown in Figure 4.

Another challenge in developing GSMS was creating a 3-D model of the Earth that could be rendered quickly enough to maintain smooth animation in a display. GSMS implements its world map data base as a series of 3-D land mass tiles that, drawn edge to edge, render the Earth as a faceted sphere. Using clipping algorithms, GSMS culls out tiles that will not appear in a scene, saving rendering time. To further accelerate the process of depicting the Earth, GSMS simulates the ocean by drawing a simple blue disk behind the land masses.

Perhaps the biggest challenge in implementing GSMS was verifying the accuracy of the displays. Before GRO launch only limited simulated data was available, and was not sufficient enough to verify simultaneously the position and attitude of GRO relative to the positions of the Sun, Earth, and Moon. It was only possible to verify the computation of GRO's attitude and the Sun and Moon positions. This was accomplished using validated test cases furnished by the FDF and data from the U.S. Naval

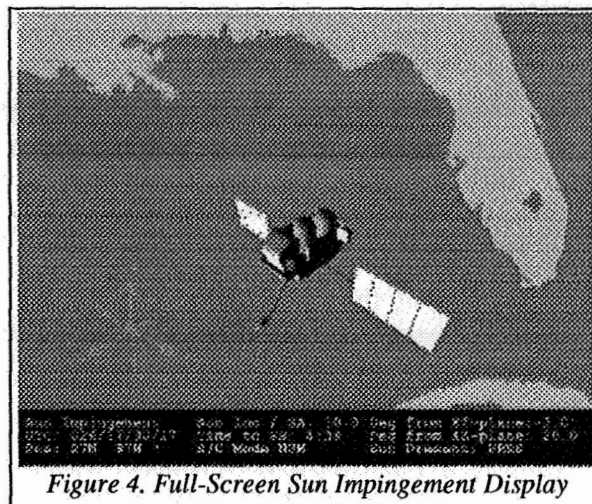


Figure 4. Full-Screen Sun Impingement Display

Observatory's ephemeris program. Immediately after launch, however, CSC and GRO spacecraft analysts were able to verify the GSMS displays by comparison to actual telemetry data.

5. IMPLEMENTATION

Because of its powerful 2-D and 3-D color graphics display capabilities, a newly available (in 1989) SGI Personal IRIS 4D/20 workstation was chosen to implement GSMS on. As required by the CCSB, software for accessing telemetry data was implemented on a 386-class PC workstation, identical with those used in the GRO MOR. To promote system portability and reusability, all GSMS software was implemented using the C programming language.

In the first year of development, several tools and procedures were implemented to facilitate the implementation of the GSMS displays. The first of these was a tool for creating an accurate 3-D model of the GRO spacecraft, implemented using Control Automation's ModelMate CAD software. This tool enabled spacecraft components to be easily digitized from blueprints and processed into a format for efficient rendering by GSMS. To build a 3-D world data base for rendering the Earth as a solid model (instead of as a wireframe image), a 3-D graphical editor was developed that allows the user to interactively generate 3-D polygonal tiles of land masses from a basic latitude/longitude world map. Another tool was built to scale each GSMS display into an icon for use in the display selection menu.

The essential components of GSMS are the numerous algorithms that produce the displays. To

compute the positions of the Sun, Moon, and stars, existing FDF algorithms were converted from Fortran to C. For developing FOV displays, a camera model capable of viewing angles as wide as 170 degrees was implemented. To simulate the FOVs of the star trackers, Sun sensors, bright object sensors, and other items of interest, this camera model was simply placed at the proper location on the spacecraft and pointed with a line-of-sight vector. Figure 5 shows GRO's high gain antenna FOV (the center circle in the view outlines the FOV of the antenna). Other significant algorithms

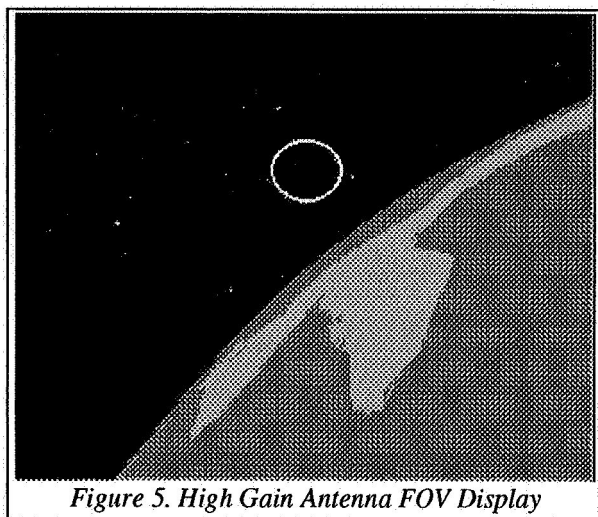


Figure 5. High Gain Antenna FOV Display

developed include those for calculating GRO's position and predicted orbit path, day/night termination line (both at ground level and at GRO's altitude), and procedures for using multiple coordinate reference frames simultaneously (e.g., Earth-centered inertial, latitude/longitude, spacecraft body, high-gain antenna, and solar array panel).

A system communication scheme synchronizes all of the separate GSMS processes. As each set of telemetry data is extracted by the PC workstation process, it is transmitted to the SGI workstation via an RS-232 serial line. Between the communicating processes on each workstation, an appropriate handshaking protocol helps ensure data integrity. On the SGI workstation, the executive communication process spawns a user interface process and a GSMS display management process. Communication between these three processes is coordinated through semaphores and shared memory. By using a double buffer technique, GSMS ensures that all telemetry data has been received before updating its displays.

6. RESULTS AND FEEDBACK

With GSMS operational, the GRO FOT began relying increasingly on the system to immediately see GRO's position and its orientation to the Sun. In monitoring GRO, knowing exactly when the spacecraft will enter or leave sunlight is important, because activity in the GRO power subsystem changes markedly then. With GSMS, FOT personnel can directly see GRO leaving or entering sunlight, and can also view a countdown clock showing the exact number of minutes and seconds to sunrise or sunset at the bottom of the display.

Whenever GRO maneuvers to a new attitude, a biweekly event, being able to see the maneuver as it occurs using GSMS is immensely reassuring to the GRO attitude analysts. They can easily verify that the highly critical roll, pitch, and yaw changes are being executed by GRO's attitude control subsystem exactly as planned. The consensus of GRO FOT personnel is that GSMS is a spacecraft monitoring tool without which they cannot operate efficiently. It has become invaluable in assisting them in rapidly understanding events that cannot be understood or explained through text displays alone.

7. SPACE VIEWS: THE NEXT GENERATION

With the enthusiasm of the GRO FOT establishing GSMS as an unqualified success, the CCSB has now tasked CSC to port GSMS to the Transportable Payload Operations Control Center (TPOCC). TPOCC is a new ground support system architecture, based on Unix workstations connected via Ethernet and operating in a client/server relationship, with display and user interface processing performed using the X Window and Motif toolkits. This new version of GSMS for TPOCC, Space Views, uses object-oriented analysis and design techniques and the C++ programming language.

Whereas GSMS was specifically tailored to meet the requirements of the GRO FOT, Space Views is being designed to have a generic framework to which specific display components can be easily added and customized for a particular spacecraft. To achieve this goal, a decision was made to use object-oriented (OO) technology. With an OO approach, what is generic and what is mission specific falls out naturally from the OO analysis. The end goal is a tool kit full of objects from which Space Views displays can be built.

8. OBJECT-ORIENTED APPROACH

As has been frequently said, object-oriented programming (OOP) represents a new paradigm shift, a new way of organizing a solution to a problem. What makes the OO approach attractive are the improvements in system robustness, efficiency, and maintainability that it promises. But to achieve these promises, there is a learning curve that must be tackled through training.

Space Views developers initiated themselves into the world of OOP by attending two major conferences, SCOOP and OOPSLA, that focus on the entire life cycle of OOP. Based on the seminars at these conferences and a survey of books on OOP, a self teaching plan for OOP was developed as a large training budget did not exist. As the development team was already proficient in C programming, C++ was selected as the OOP language for implementing Space Views, the only other viable alternative being Smalltalk. The widespread availability of C++ support tools on a variety of platforms was also a factor in the decision to choose it for implementing Space Views. But before delving into C++ programming, it was necessary to put everyone into the OO way of viewing the world.

Endorsed by leading practioners of OOP was the Class, Responsibilities, and Collaborators (CRC) notecard method of object-oriented analysis (OOA). This was practiced on small sample problems in several highly interactive 2-hour sessions. The goal here was to get everyone to think like an object. At the same time, the leading books on OOP were acquired, but the ones having the best readability were An Introduction to Object-Oriented Programming by Timothy Budd, Object-Oriented Design by Grady Booch, and Object-Oriented Analysis (2nd Edition) and its companion, Object-Oriented Design by Peter Coad and Ed Yourdon. CSC has selected the Coad and Yourdon OOA tool program to document the Space Views OOA activity, and plans to use their OOD tool when it becomes available. An OOA Training video by Peter Coad has also been acquired and will be used to assist management personnel with understanding the Coad and Yourdon OOA methodology and notation.

After several weeks of OOA execercises, training was started on C++ through the C++ Primer (2nd Edition) by Stanley Lippman, the PC-based, self paced, Teach Yourself C++ (2nd Edition) by Al

Stevens and, of course, The C++ Programming Language (2nd Edition) by Bjarne Stroustrup.

Another hurdle to overcome has been learning how to use X Window and implement displays as Motif widgets. Most of the learning has been through on-the-job trainng, working with TPOCC personnel already involved with these activities. Additional help has come from books, and from video courses on X and Motif available at GSFC. A prototype Space Views map display has already been built as a Motif widget, and rapid prototyping will continue to be used in parallel with the Space Views analysis and design activities.

Since Space Views has 3-D as well as 2-D displays, initially there was an issue of whether to use the PHIGS (an ANSI and ISO 3-D graphics programming standard) extension to X (PEX) graphics library/application program interface (API). PEX is an evolving standard and current vendor implementations of it exhibit, at least in our experience, inconsistant rendering behavior. Adopting use of PEX implied adding risk to the development of Space Views. Recently Silicon Graphics anounced GLX, the counterpart to PEX, that allows applications built with SGI's graphics language (GL), such as Space Views, to run under X. It is hoped that GLX will resolve this issue of implementing 3-D displays under X, since GL is now being made available to all major workstation vendors in an open architecture arrangement (OpenGL). Using GLX represents less risk than using PEX since the Space Views development team has several years of programming experience with GL, having implemented GSMS using GL, but only minimal experience with PEX, which was explored by implementing a small prototype application.

9. SUMMARY

GSMS proved that FOT personnel need direct visualization of spacecraft operations, a real-world view of events, and not just numbers updating on a screen. By the end of 1993, Space Views is expected to be in place supporting the SAMPEX mission, and ready to assist upcoming missions. At GSFC, Space Views will be the visualization tool for supporting the next generation of modern spacecraft. Space Views will use OOP to maximize efficiency, configurability, and reusability while minimizing future development and maintenance costs.

N94-23891

VMPLOT: A VERSATILE ANALYSIS TOOL FOR MISSION OPERATIONS

Allen W. Bucher, Owen G. Short

Martin Marietta Astronautics Group, Denver, Colorado 80201, U.S.A.

ABSTRACT

VMPLOT is a versatile analysis tool designed by the Magellan Spacecraft Team to graphically display engineering data used to support mission operations. While there is nothing revolutionary or innovative about graphical data analysis tools, VMPLLOT has some distinguishing features that set it apart from other custom or commercially available software packages. These features include the ability to utilize time in a Universal Time Coordinated (UTC) or Spacecraft Clock (SCLK) format as an enumerated data type, the ability to automatically scale both axes based on the data to be displayed (including time), the ability to combine data from different files, and the ability to utilize the program either interactively or in batch mode, thereby enhancing automation.

Another important feature of VMPLLOT not visible to the user is the software engineering philosophies utilized. A layered approach was used to isolate program functionality to different layers. This was done to increase program portability to different platforms and to ease maintenance and enhancements due to changing requirements. This paper will describe the functionality of the unique features of VMPLLOT as well as highlighting the algorithms that make these features possible. The software engineering philosophies used in the creation of the software tool will also be summarized.

Keywords: mission operations, data analysis, plotting software, Magellan.

1. INTRODUCTION

The Magellan Spacecraft is the National Aeronautics and Space Administration's planetary mission designed to obtain a high-resolution Synthetic Aperture Radar (SAR) image of the surface of Venus. Launched from the Space Shuttle Atlantis on May 4, 1989, Magellan traveled one-and-one-half times around the Sun to arrive at Venus on August 10,

1990. SAR imaging of the Venusian surface commenced on September 15, 1990 revealing the most astounding surface views ever recorded.

The Magellan Spacecraft Team, comprised of a group of specialized spacecraft engineers, is responsible for monitoring the day-to-day health of the Magellan Spacecraft. These engineers utilize a variety of software tools to assess the health and status of each spacecraft subsystem. The most widely used tool by the Magellan Spacecraft Team is VMPLLOT. VMPLLOT is a software program designed to produce hard copy and screen plots of user selected data.

2. VMPLLOT PROGRAM OVERVIEW

VMPLLOT is a data file driven program designed to allow Magellan engineers to graphically display spacecraft data on a Visual Display Terminal or Hard Copy device. The main display devices currently supported by the program include the Sun Workstation "gfxtool", X Windows, and the Tektronics 4014 display terminal or Tektronics 4014 emulator. The Hard Copy devices supported are "Laser Writer" type devices that are capable of interpreting postscript files.

VMPLLOT receives user data from two required sources; a DATA file and a CONTROL file.

The DATA file(s) are typically time ordered files that contain the x/y data pairs that the user wishes to plot. These files are formatted as shown in Figure 1. The first column is reserved for the time column, either a Universal Time Coordinated (UTC) formatted "time tag" or a time tag formatted in the mission specific Spacecraft Clock (SCLK) format. The other columns are used to contain the user data; spacecraft voltages, for example. As a generic constraint the DATA file is restricted to a single time column followed by up to 36 data columns. The format of these files are fixed and documented in a formal software interface specification.


```

tBAT1      SCET 03 Bat1 Voltage Bat2 Voltage MainBus Volt
92-206/00:00:00.000 EU Vdc      EU Vdc      EU Vdc
92-206/23:59:59.999 %12.4f     %12.4f     %12.4f
00090 92-224/23:03 E-0174      E-0175      E-0180
92-206/00:03:59.424      31.1410      31.1228      31.2815
92-206/00:04:59.424      31.1410      31.1228      31.2815
92-206/00:05:59.424      31.1410      31.1228      31.2815
92-206/00:06:59.424      31.1410      31.1228      31.2815
92-206/00:07:59.424      31.1410      31.1228      31.2815
92-206/00:08:59.424      31.1410      31.1228      31.2815
92-206/00:09:59.424      31.1410      31.1228      31.2815
.
.
.
. (or SCLK)
01600591:40      31.0002      30.9815      31.1401
01600592:40      31.0002      30.9815      31.1401
01600593:40      31.0002      30.9815      31.1401
01600594:40      31.0002      30.9815      31.1401
01600595:40      31.0002      30.9815      31.1401
01600596:40      31.0002      30.9815      31.1401
01600597:40      31.0002      30.9815      31.1401

```

Figure 1 VMPLLOT DATA File

The CONTROL file instructs VMPLLOT which data to plot and how to plot it. The CONTROL file, as shown in Figure 2, contains the DATA file name(s), the plot start and end times, the data columns to plot, the plot axis information, the plot line format, and the plot titles.

The main structure within the CONTROL file is the \$SOP (Start Of Plot) - \$EOP (End Of Plot) blocks that control the production of a single plot image. Each \$SOP-\$EOP block must contain exactly 26 records including the \$SOP-\$EOP records. Each line of the \$SOP-\$EOP block has special meaning to VMPLLOT and must be used in conjunction with other lines within the block to achieve accurate results. The selected CONTROL file may contain any number of \$SOP-\$EOP blocks. Each block results in the production of one plot image. This feature allows trending plots for related components to be grouped in a single CONTROL file to aid the automation of plot production.

2.1. Plot Types

The two basic plot types utilized by VMPLLOT are Channel versus Time and Channel versus Channel. A channel is defined as a column of data from a DATA file. In spacecraft operations most of the data in the DATA file originates from telemetry channels, hence the name.

```

$SOP /* PLOT #1 */
2      ! Number of DATA files
tickmap.drf      ! DATA file 1 name
motorcur.drf      ! DATA file 2 name
1      ! Type 1,2=CVT 3,4=CVC 5,6=REL 7,8=STRP
89-348/20:00:00.000      ! Start Time of the plot
89-348/20:25:30.000      ! End Time of the plot
00      ! Column for x data - file1
00      ! Column for x data - file2
1      ! Num y-axis variables - file1
2      ! Num y-axis variables - file2
01 00 00 00      ! Col nums of y variables - file1
02 03 00 00      ! Col nums of y variables - file2
AUTO      ! X axis scale type AUTO or MAN
N/A      ! X Axis Min, Max (MANual scale)
N/A      ! # Maj Ticks, # Minor/Major Ticks
1      ! Draw an X-Axis Grid 1=Yes, 0=No
MAN      ! Y axis scale type AUTO, MAN, LOG
0 250      ! Y Axis Min, Max (MAN scale), Decade
6 4      ! MajTicks & MinorTicks, or Start Exp
1      ! Draw an Y-Axis Grid 1=Yes, 0=No
2      ! Line Format 1=strip,2=line,3=points
DMS A Motor Current - Radar Functional      ! Plot Title
SCET      ! X Axis Title
Current in Milliamps      ! Y Axis Title
$EOP
$SOP /* PLOT #2 */
1      ! Number of DATA files
tBAT1.drf      ! DATA file 1 name
.
.
.
$EOP

```

Figure 2 VMPLLOT CONTROL File

2.1.1. Channel Versus Time

A Channel versus Time (CVT) plot, as shown in Figure 3, includes one or more data channels plotted versus time. The time axis may either be a UTC or SCLK formatted time tag. The UTC time tags are formatted YY-DDD/HH:MM:SS.FFF where YY is the last two digits of the year, DDD is the day of the year, and HH:MM:SS.FFF is hours, minutes, seconds, and fractional seconds. The SCLK time tag is formatted RIM:MOD91 where the Real-Time Image Count (RIM) is a 24 bit counter which is incremented every 60-2/3 seconds and the Mod 91 Count (MOD91) is an 8 bit counter which increments every 2/3 of a second. Both of these time formats are processed by VMPLLOT to allow spacecraft analysts to view spacecraft events versus time. The majority of spacecraft analysis during Mission Operations is accomplished by viewing spacecraft performance as a function of time (i.e. spacecraft trends over time).

2.1.2. Channel Versus Channel

A Channel versus Channel (CVC) plot, Figure 4, includes a data channel plotted against another data channel. The most common use of this plot type for Magellan was the analysis of attitude phase plane information (spacecraft position versus rate) and the analysis of High Gain Antenna signal strength versus spacecraft offset. This feature was also used to plot data that is not associated with a specific event time.

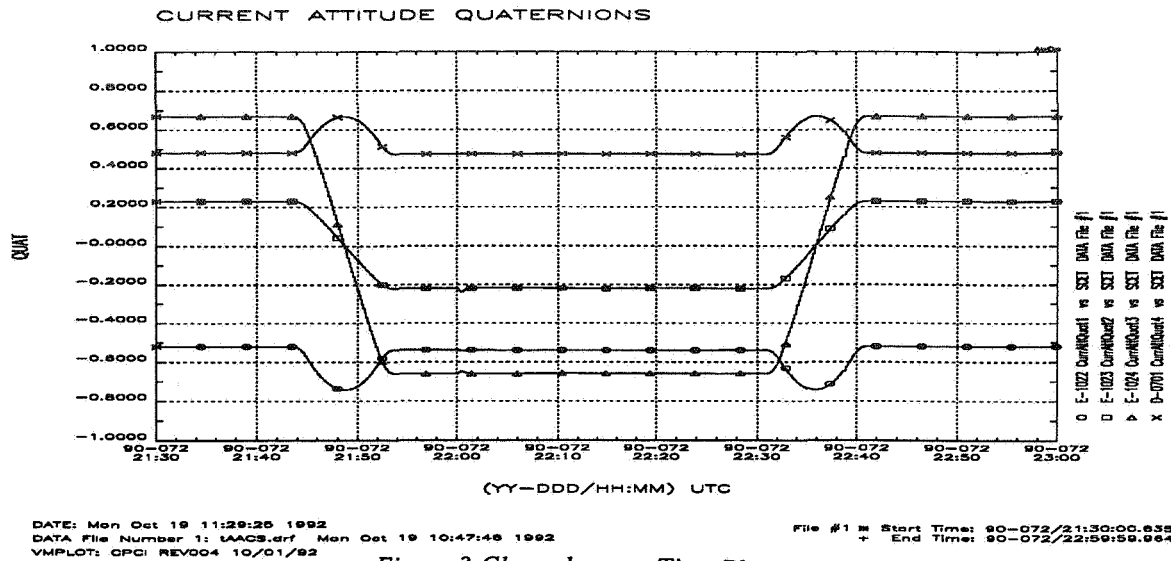


Figure 3 Channel versus Time Plot

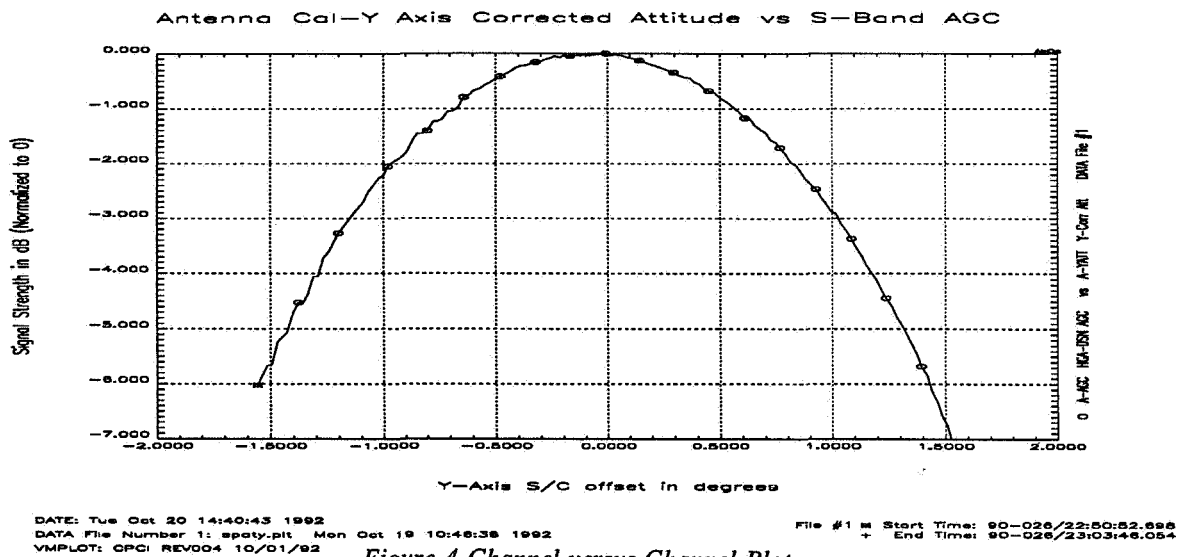


Figure 4 Channel versus Channel Plot

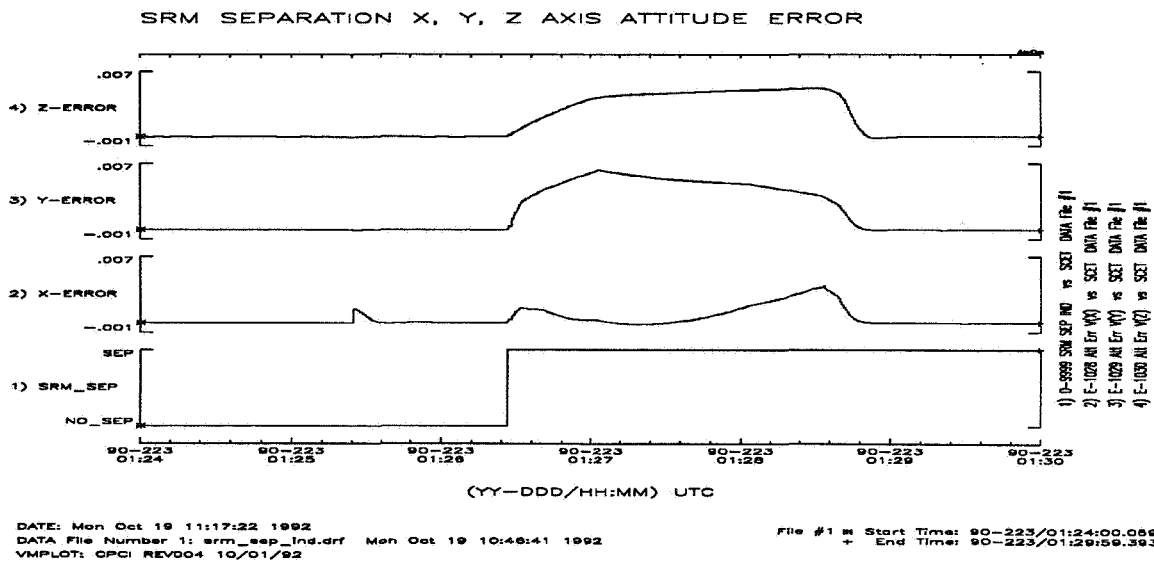


Figure 5 Strip Chart Style

2.1.3. Overplotting and Strip Charts

In combination with the two main plot types described above, CVT and CVC, VMPLLOT provides three methods to view the data. The default method provided is the single channel plot, either a single channel versus time or a single channel versus another channel, Figure 4. The second method, as illustrated in Figure 3, is overplotting of up to four channels versus time or up to four channels versus a single channel. In addition to the methods described above, another method is the production of up to four strip chart type plots versus time, Figure 5.

3. VMPLLOT FUNCTIONALITY

VMPLLOT was designed and coded to meet the requirements of the Magellan Spacecraft Team. The Spacecraft Team levied requirements including the plot template outline, the different plot types, the scale types and options, the drawing types, the display requirements for the data, the devices that the program must support, and the program user interface.

The flow of the VMPLLOT program contains many features and functions. Although all features of the software are important, the following paragraphs will highlight the unique features of the program. These features include the ability to utilize Universal Time Coordinated or Spacecraft Clock times as enumerated data types, the ability to automatically scale both axes, and the the ability to combine data from different files on the same plot image. The algorithms behind the features will also be highlighted, where appropriate.

VMPLLOT uses a standard Cartesian Coordinate system for its axes and plotting region defining the y values as the ordinate values and the x values as the abscissa values. The program is designed to allow the X Axis to support either time or engineering values. The Y axis can be scaled to support linear scale or logarithmic scale types.

3.1. Time Scaling

The most distinguished feature of VMPLLOT is its ability to utilize time as an enumerated data type. The time types currently supported for Magellan include Universal Time Coordinated (UTC) representing Spacecraft Event Time (SCET), Earth Received Time (ERT), and Monitor Sample Time (MST) and Spacecraft Clock (SCLK), see section 2.1.1. VMPLLOT processes time, either UTC or SCLK, by first converting it to a common unit, decimal seconds. For UTC this is accomplished by

converting the time tags to decimal seconds since 1980. For SCLK the the two main clock components (RIM and MOD91) are converted to seconds and fractions of seconds and combined to form a single decimal second representation.

3.1.1. UTC Scaling Algorithm

For UTC axis the total time delta in minutes is determined from the difference between the plot start and end times. Although the common time unit is decimal seconds, the plot scales are determined with the minimum resolution of one minute. The time delta is utilized to determine the appropriate intervals between successive "ticks" on the axis scale. The "tick" interval is selected from one of twenty-three predefined tick divisions ranging from a minimum of one minute to a maximum of 576,000 minutes (400 days) between successive ticks on the scale. Once the tick interval is determined, the minor (non-labeled) ticks are drawn. After the minor ticks are completed, the first "nice" interval is determined by finding the first whole increment of time, hours or minutes depending upon the resolution of the scale. The program then places the first axis label on the image. The remainder of the axis labels are then drawn at the predefined interval. The plot start and end times in seconds are then utilized to setup the scaling algorithms to assure the data is correctly plotted against the scale just determined. See Figure 3.

3.1.2. SCLK Scaling Algorithm

Since the SCLK time in seconds is similar to other floating point numbers processed by VMPLLOT, the SCLK time in decimal seconds is processed by the auto scaling routine, section 3.2, to determine the scale maximum and minimum values. This information is then passed to the routine for drawing normal engineering axes and the axis is treated as any other engineering number, except that the labels are placed and labeled according to the Magellan SCLK conventions.

3.2. Automatic Scaling Algorithm

The automatic scaling routine is designed to accomplish two goals for engineering scales; first determine the scale boundaries based on the input data to be plotted, and second determine the axis labels and intervals to guarantee that the axis labels are at even intervals (Ref. 1). Based on the maximum and minimum values from the input data set, the automatic scaling routine determines the number of label ticks to be placed on the scale, the delta between each label, and the scale maximum and

minimum. The scale maximum and minimum values are then utilized to determine the scaling algorithms to assure the data is correctly plotted.

3.3. Plotting Data

After determining the plot axis and scales, the user data x/y pairs are then "drawn" to the plot image. The world plotting coordinate bounds used by VMPLLOT are fixed. The program determines scaling factors between user and world coordinates, slope and intercept, based on the input user data set x and y axis scaling information. Once the scaling factors are determined, each x/y user data pair is scaled to fit within the predetermined world coordinate bounds. The world coordinates of the x/y data pair to be plotted are determined by the equation:

$$w = \text{slope} \times \text{user_data} + \text{intercept}$$

VMPLLOT plots the user data utilizing poly line or poly marker segments of one thousand data points each. Each plot image can contain from 1 to n segments of 1000 data points. This means that there is no program limitations (other than host machine limitations) regarding the maximum number of x/y data pairs that can appear on a single plot image. Plot images with in excess of 100,000 x/y data points have been processed by VMPLLOT. The plotting routine also identifies the first x/y pair plotted with a star symbol and the last x/y pair plotted with a cross symbol. The times associated with these data points are also recorded and displayed to the plot image as an indicator of the actual data start and end times.

For the overplotting and strip charts display methods, described in section 2.1.3, the data to be plotted may be selected from one DATA file or from two DATA files. This characteristic allows VMPLLOT to extract one set of x/y pairs from one file and another set of x/y pairs from another file and overplot all x/y pairs on the same image. In all cases however, x/y pairs must come from the same DATA file. This feature was created to allow engineers to compare actual spacecraft events versus events predicted by ground software models.

4. SOFTWARE ENGINEERING

The Software Engineering philosophy used in developing VMPLLOT was a combination of *Structured Programming* and *Layered Architecture*.

Structured Programming was formulated in the late '60's and early '70's. It is a discipline for programming which emphasizes the use of only one entry and exit point per construct. It is described in Refs.

2-4 as programming using only sequence, selection, and iteration statements in place of gotos for program control. Any program written using goto statements can be rewritten using if-then-else, while, and sequence statements. Although the main emphasis of structured programming is programming using well defined constructs, it also includes top down design and repeated subdividing of major tasks into subtasks. The subtasks must have a clear objective that is easy to design, code, and test.

The main idea of *Layered Architecture* is isolating different functionalities to different layers. This approach is similar to the underlying principles of the International Standards Organization Open Systems Interconnect (or ISO/OSI) reference model. In a layered approach, each layer is its own independent entity. It has no knowledge of the processing details of the other layers. Each layer knows only of its tasks and what it must provide to or receive from adjacent layers. By concentrating on designing VMPLLOT using this approach, each different function can be isolated to its own distinct layer. This layering also allowed the software engineer to isolate the environment or host dependent functions, allowing the software products to be used in different computing environments with minimal impact on coding and documentation efforts.

By utilizing both structured programming and layered architecture, software systems can be developed which emphasize portability and flexibility. By approaching software development utilizing these techniques and carefully designing the system, software can be written and easily transported between different environments. This design process builds upon the ideas put forth by Kernighan and Ritchie in the standardization of C to provide maximum portability of code (Refs.5-6). Although code portability plays an important role in this structured and layered technique, the software engineering approach being considered treats the software development process consisting of Requirements, Design, Code, and Test as a whole. This approach also allows all products of the lifecycle to be flexible enough to entertain changes without major impacts.

The Magellan Project Team realized cost savings from these philosophies when the program was ported to other host machines with minimal effort in support of Assembly, Test, and Launch Operations and System Verification Laboratory efforts. This program has also been ported for use to support other programs including the Transfer Orbit Stage and Mars Observer (xvmplot).

4.1. VMPLLOT Implementation

The structured and layered architecture of VMPLLOT evolved during the design phase. The program was divided into three layers, illustrated in Figure 7, consisting of Data Manipulation Routines (Layer 2), Graphics Package Interface Routines (Layer 1), and Vendor Graphics or Device Interface Routines (Layer 0). The layer 2 routines control all user and data interaction to the program. The layer 1 routines control all graphics calls to the graphics package being utilized. The layer 0 routines contain the actual graphics primitives being utilized, either a commercial package or user defined routines, to produce the graphic effect.

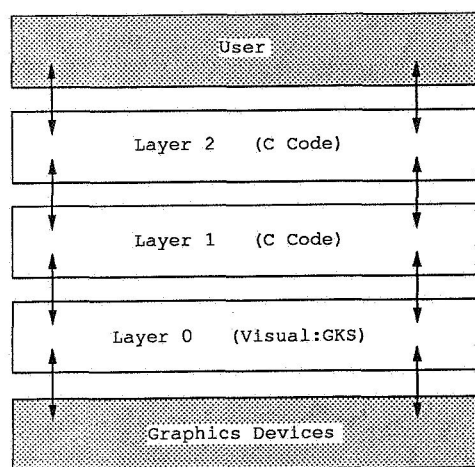


Figure 7 VMPLLOT Functional Layers

VMPLLOT was coded using the C programming language. The system code was developed using a top-down approach, adhering to the structured and layered constructs introduced during the design phase. The layer 2 and layer 1 routines were custom routines written specifically to satisfy the unique VMPLLOT software and data manipulation requirements. The layer 0 routines were provided by purchasing the commercial software package Visual:GKS, developed by Visual Engineering.

Visual:GKS was selected because of its ability to easily satisfy the three main graphics primitives required by VMPLLOT; the ability to move the current position within the plotting surface, the ability to draw a line, and the ability to place text on the plot image. Another key feature of Visual:GKS is the ability to produce graphic output for any number of devices utilizing the GRAPHCAP technology. The GRAPHCAP technology, similar to the Unix TERMCAP technology, allows VMPLLOT to drive many different output devices with only

minimal code changes for device identification. This concept allows the layer 2 and layer 1 routines to be virtually device independent.

5. SUMMARY

The VMPLLOT program was designed specifically to support spacecraft data analysis in a Mission Operations Environment. The majority of spacecraft data analysis is accomplished by analyzing or trending events over time. Another characteristic of Mission Operations is that the daily tasks tend to be repetitive in nature. Knowing these facts, VMPLLOT was designed to support the time oriented analysis of spacecraft engineering data and provide ease of operation to simplify the daily burden of data analysis. VMPLLOT is a generic versatile data analysis tool capable of supporting spacecraft Mission Operations.

6. REFERENCES

1. Lewart, Cass R., "Subroutine plots data on graph automatically", *Electronics*, July 14, 1983, pages 148-151.
2. Mills, Harlan D., and Linger, Richard C., "Data Structured Programming: Program Design without Arrays and Pointers", *IEEE Transactions on Software Engineering*, vol. SE-12, no. 2, February 1986, pages 192-197.
3. Linger, R.C., Mills, H.D., Witt, B.I., *Structured Programming: Theory and Practice*, Reading, MA: Addison-Wesley, 1979.
4. Dahl, O.J., Dijkstra, E.W., Hoare, C.A.R., *Structured Programming*, New York, NY: Academic Press INC., 1972.
5. Kernighan, B.W. and Ritchie, D.M., *The C Programming Language*, Englewood Cliffs, N.J.: Prentice-Hall, 1978.
6. Johnson, S.C. and Ritchie, D.M., "Portability of C Programs and the UNIX System", *The Bell System Technical Journal*, Vol. 57, no. 6, July-August 1978, pages 2021-2048.

7. ACKNOWLEDGEMENT

The work described in this paper was accomplished at Martin Marietta under contract to the Jet Propulsion Laboratory, California Institute of Technology and sponsored by the National Aeronautics and Space Administration.

Unix is a Trademark of Bell Laboratories.

Visual:GKS is a Trademark of Visual Engineering, Inc.

REAL-TIME AUTOMATED FAILURE IDENTIFICATION IN THE CONTROL CENTER COMPLEX (CCC)

N94-23892

Sarah Kirby, Janet Lauritsen, Ginger Pack
NASA Johnson Space Center

Anhhoang Ha, Steven Jowers, Robert McNenny, The Truong
McDonnell Douglas Space Systems Company, Houston Division

James Dell
Loral Space Information Systems

ABSTRACT

This paper describes a system which will provide real-time failure management support to the Space Station Freedom program. The system's use of a simplified form of model based reasoning qualifies it as an advanced automation system. However, it differs from most such systems in that it has been designed from the outset to meet two sets of requirements.

First, it must provide a useful increment to the fault management capabilities of the Johnson Space Center (JSC) Control Center Complex (CCC) Fault Detection Management system. Second, it must satisfy CCC operational environment constraints such as cost, computer resource requirements, verification and validation, etc. The need to meet both requirement sets presents a much greater design challenge than would have been the case had functionality been the sole design consideration. This paper overviews the choice of technology, discussing aspects of that choice and the process for migrating it into the control center.

1. INTRODUCTION

This paper provides an overview of the underlying technology and design of Extended Real-Time FEAT (ERF) within the context of its migration to the CCC. Section 2 describes the fault detection and identification (FDI) problem, specifically focusing on those functions ERF now addresses and how we arrived at this focus. Section 3 overviews the constraints imposed upon the developer - both those imposed because of good software engineering and those imposed by the nature of the environment. Section 4 examines the choice of technology, the algorithms developed, and possible extensions and adjuncts to it. Section 5 overviews the solution settled upon, its current status and general comments. Section 6 finishes the paper with concluding remarks.

2. FDIR PROBLEM

2.1 The FDIR Task

From the start, this project has not sought to force a particular technology into the operations environment but rather to understand the environment and the task that the end-user must accomplish. Specifically, we addressed the fault detection, identification, and recovery (FDIR) aspects of systems monitoring and control. We sought to understand the FDIR task from the Mission Controller (MC) perspective without regard to how those tasks were to be done (i.e., by human or machine). A task analysis was performed which spanned several months and was informally documented (Ref. 1). It involved mission controllers, NASA and contractor management, and technology experts. Following this analysis, the group decided how to partition the tasks between human and machine. There were clearly some tasks which could be automated immediately and others which must be left to the MC. Still others were subjects for subsequent automation. (Since the most complex defined task is recovery, the familiar FDIR will appear henceforth as FDI.)

2.2 The Subtask To Automate

The task analysis defined three types of failures. Explicit predefined faults were defined as "those for which specific malfunction and recovery procedures are developed and validated. Typically, explicit predefined faults are time critical." Implicit predefined faults were defined as "those which are understood and for which conceptual malfunction and recovery procedures exist, but written procedures have not been developed and validated. For example, malfunction and recovery procedures normally do not address degraded systems (i.e., systems in which a failure has already occurred)." Undefined faults are those for

which no *a priori* mappings exist for the signature/fault pair.

From the task analysis came a number of subtasks which the controller must perform for fault analysis, regardless of the fault type (explicit, implicit, or undefined). After examining the capabilities which the then baselined control center was to provide, we concluded that CCC baseline support existed only for explicit, predefined failures. An opportunity to expand support to include implicit failures seemed to present itself for specific failure analysis functions. These were commonality assessment among multiple alarms and/or out-of-limit conditions, identification of failures, and impact assessment. The consensus was that automating analysis of undefined faults is a research topic.

2.3 ERF's Role

Other subsystems of the CCC Fault Detection and Management (FDM) system interpret onboard Caution and Warning (C&W) codes, generate and display onboard Fault Summary Messages (FSM) and Ground Detected Onboard Fault (GDOF) messages, issue alerts and audible alarms, log FSM and GDOF messages, and pass data to the ERF subsystem for further processing. ERF augments the core FDM capability by analyzing explicit and implicit predefined faults (note: a representation of implicit faults also captures the explicit faults as a subset). ERF will be handed a problem definition (failed conditions) and after gathering other instant state information (e.g., sensors known to be within limits, degraded system state, etc.) perform an automated analysis, always under the MC's oversight and control.

ERF's capabilities can be conceptually broken into two parts; fault analysis and problem management. Fault analysis consists of Commonality Assessment, Failure Identification and Impact Assessment. The problem management functions include management of queues if multiple problems are to be worked, queuing of analysis commands, interfacing with external entities, interrupt/resume functions, etc. At any time during processing the MC can interrupt the process and enter a "hypothetical" mode of operation. The systems state information can also be used to populate the model with an initial state and the MC can then explore "what if?" scenarios based upon current, observable conditions.

3. CONSTRAINTS

ERF has been designed with the CCC as its target environment from its genesis. We have been extremely careful not to use techniques or make assumptions which would preclude its use in the CCC. We acknowledge that this may result in an

approach which may be defined too conservative by those whose responsibilities are to develop and understand new technology. However, a flight-critical application by its very nature lies outside the scope of a research lab. Real-world constraints must be considered: robust interfaces to real-time data must be provided, missing and noisy sensor data must be dealt with, multiple problems must be queued, analyses must be interrupted and resumed, etc. Real-world applications need also to consider end-user constraints and implementation requirements.

3.1 End-User Constraints

MC concerns posed a number of constraints on the design of ERF. Broad user requirements are documented in (Ref. 2). ERF's user requirements are either an explicit subset of these or are derived therefrom. These include requirements for:

- Compatibility with proven real-time mission support operations
- Prompt solution response
- Positive MC control at all times
- Generic FDI capability.

Any automated system designed for real-time mission operations must support the process which human controllers currently use. MCs work in highly integrated, well trained teams. Shifting part of a team's responsibility to a computer means that the computer must play a part in the team; it cannot be a solo player (Ref. 3).

Accurate identification of the failure at hand is a prerequisite for dealing with critical situations. This means that to be useful when it is really needed, an automated FDI system must take no longer to provide a solution than would a highly experienced MC. ERF is required to identify failures and assess their impacts within 5 minutes on average, 6 in the worst case.

Just as a human team player must take orders, ERF must always be directly controllable by the MC. This means that ERF must keep the MC informed of its intentions and allow him or her to redirect it, or to halt it if necessary, at any time.

Since failure identification is in principle a generic process, ERF is required to support FDI for all Space Station onboard systems for which its basic knowledge representation scheme is appropriate.

3.2 Implementation Constraints

Any development of an application whose target is within the real, operational world needs to satisfy

such real-world constraints as cost ceilings, resource allocation limits, and fast algorithm performance. Sometimes data will be noisy and/or missing. Interfaces with other CCC systems (Status and Control, database functions, the data streams, etc.) must exist. Choice of platform, development methodology and language will be made by the control center developers, not the technologists. The application and the models it uses must be susceptible to verification, validation, and maintenance.

In the case of the CCC, the constraints included the use of Ada for all new code, minimal cost, working within the constraints of the development environment for the CCC, following methodological guidelines for formal software development, and formal reviews.

4. THE TECHNOLOGY

4.1 Its Description

ERF is fundamentally built upon a bi-valued model representation. The models are causal networks of failure represented as directed graphs. They are automatically configured via telemetry and knowledge of system degradation to represent the system's observable state. These models are then operated over to infer information about commonality among annunciated out-of-limit or alarm conditions, what could have been the cause or causes of these annunciations, and to where these effects might propagate.

The Failure Environment Analysis Tool (FEAT), built by the JSC Intelligent Systems Branch, computes transitive closure for the given model, displaying the results of queries graphically. For details, see (Refs. 4-6). ERF is a layer of software on top of FEAT to extend its capabilities. Specifically, ERF sequences queries to FEAT in a manner similar the way a MC might use it to analyze either real-time data or hypothetical data. The results are displayed on FEAT's displays and/or on displays external to FEAT.

Currently, FEAT runs on both Macintosh and Unix platforms (Refs. 4-5). A companion product, the Digraph Editor (Ref. 4), provides an environment to aid in constructing failure models for FEAT, though other tools which adhere to the PICT standard can produce working digraphs for FEAT. Both products are written in C and are available for the Macintosh through COSMIC. The Unix version has just been submitted to the Space Station Freedom Program's Technical and Management Information System (TMIS) and is now available to the Space Station Program. Eventually it will be available to all through COSMIC as well. Table 1 (at the end of this paper) lists FEAT capabilities and the enhancements that ERF will provide.

4.2 Rationale

The technology was selected because it provides the required functionality and satisfies the constraints:

- It utilizes relational failure models of the systems being analyzed.
- The digraph failure models documented via a graphical representation are more easily maintainable than equivalent text representations.
- It utilizes a standard mathematical technique for computing reachability (connectivity) analyses; e.g., transitive closure.
- In addition, it utilizes heuristic knowledge about how to diagnose failures in physical systems.

From an operations perspective ERF supports real-time fault analysis and facilitates hypothetical reasoning on the part of the MC. It can be used to support training and the generation of Support Products such as malfunction procedures, Failure Modes Effects Analysis/Critical Items List (FMEA/CIL), and operations procedures. Its displays can aid in providing an explanation of results by presenting a graphically expressed chain of events. It is cost effective; it uses software engineering principles to minimize life-cycle cost and is implemented in Ada around a standard software product. It will be tested using the same tools as the rest of the CCC.

ERF allows FDI applications to be developed without the system operational failure experience heretofore thought essential for developing such systems. The digraph failure model describes, on the basis of design derived knowledge, how a system *must* fail. ERF's algorithms provide a methodology for interpreting the model based on sensor data. Since the sensor interpretation algorithms are model independent, ERF will work for any system for which a FEAT-compatible failure model is provided.

4.3 Potential

The digraph representation is both concise and capable of expressing complex patterns of failure behavior. Since ERF uses models of failure behavior, derived from knowledge of system structure, and system state information, ERF provides a simplified form of model based reasoning.

ERF can incorporate increasingly sophisticated analysis techniques within its own analysis routines and/or provide initial hypothesis generation for other external algorithms such as those contained in true model based reasoning systems.

The use of the failure model, as noted earlier, supports a number of uses in the operations community. Additionally, we envision possible uses in Recovery Planning, Intelligent Computer Aided Training (ICAT), and Design Knowledge Capture (DKC). The digraph as a knowledge source can support many applications. As operational experience is acquired, models are easy to modify and heuristic information can be easily added, either as rules or implemented in conventional, procedural code.

Digraphs and their foundation can provide much more than just simple cause/effect propagation. The digraph representation is a subset of propositional logic. Consequently, if the representation were enriched to include full propositional logic, ERF could begin using models of system behavior from which both nominal and failure behavior can be analyzed using state information.

5. ALGORITHM OVERVIEW

This section overviews only at a conceptual level the analysis algorithms which ERF implements. At the highest level, these are Commonality Assessment, Failure Identification, and Impact Assessment.

5.1 ERF Core Analysis Algorithms

The Commonality Assessment algorithm uses the digraph to determine if there are paths between components having failure indications. If there are and there are no dependencies which could impede the failure propagation, then we can assert that one of the sensors is "primary" and the others downstream of it are "secondary".

Failure Identification takes the announced conditions, retrieves additional state information and categorizes the observables in the model into one of three states; good, bad, and unknown. From this, ERF builds the initial set of possible causes. This set is then pruned using knowledge about the known good components and digraph modeling artifacts to narrow the space of possible causes. Any remaining unknown observables in the model are then presented to the MC. If additional information can be obtained, the analysis is refined. For details on an early version of this algorithm, see (Ref. 7).

The Impact Assessment function takes these possible causes and predicts their respective effects on the system. ERF will annunciate: lost redundancies, where a failure may propagate if the other leg in the redundancy is lost, and new susceptibilities for critical functions (i.e., new single and dual failure sources). Other possible subfunctions include calling out which observables to monitor for indication of a failure propagating toward a critical function.

5.2 Possible Extensions/Adjuncts

ERF has been specifically designed to facilitate growth. ERF is especially *not* viewed as the panacea for failure analysis. We readily acknowledge that the digraph bi-valued model is a low fidelity model. One possible way to increase its value is to include full propositional logic capability. Even so, it would still be bi-valued and (relatively) low in fidelity. Another possible scenario is for ERF to hand its analysis off to other algorithms. We have started working with the Thermal Control System Automation Project (TCSAP) to explore how ERF might replace some of the knowledge base portions of their system. ERF's analysis results would be used as the set of initial hypothesis for a deeper analysis by a model based reasoner in that case.

6. THE SOLUTION

6.1 Development Methodology

It is important to note the design philosophy which is being used for ERF's development. The process started with the MC defining the FDIR task without respect to its allocation between man and machine.¹ Once defined, a subset of these were selected for computer implementation; limited Fault Detection and Identification and some problem management functions (queueing, interruption, etc.). The use of FEAT and the enhancements which together comprise ERF are hence couched in a broader setting, that of an integrated FDI system. While ERF does not implement all FDI requirements, its design has been influenced by an understanding that there is a "bigger picture". Effort has been made to provide hooks for swapping underlying analysis engines and the changing of internal analysis algorithms. The use of a bi-state failure model built on partial propositional logic can only provide limited insight into a system's true behavior. Hence the need to interface with other analysis applications which use alternate model representations. It is also worth noting that ERF provides automated FDI analysis for any system which can model failure as a causal network of failure modes -it is in this sense a truly generic FDI application.

In implementing these requirements, we have sought to adhere to the guidelines and milestones for CCC development. The appropriate NASA management and contractors have been involved. Funding has been switched from one directorate to another to facilitate

1. This project had no "formal" guidelines for its development methodology at its genesis. Much was based on prior experience, common sense and fortunate timing. A recent publication (Ref. 3) is an excellent reference. We strongly recommend the reading of this memo by anyone who desires their project to receive acceptance into the operational community.

technology transfer. Mission Controllers have been involved since its inception. The net effect has been a project with MC support, management approval, and contractor acceptance. This effort has involved three JSC directorates and three NASA contractors.

6.2 Status

Today, FEAT is baselined for use in the CCC and will be delivered to the CCC Testbed in December of this year. ERF had a Preliminary Design Review in March 92 (Ref. 8) and is now baselined. Its analysis algorithms are to be available in the testbed in June of 1993 and in the CCC as part of the FDM delivery in 1994.

An informal prototype for algorithm development and detail design activities currently exists. As of October 1, 1992, it consisted of two separately running processes -the GUI and the analysis routines. While far from being the system to be delivered to the CCC, the current informal prototype demonstrates:

1. the basic analysis functions of Commonality, Failure Assessment, and Impact Assessment
2. the use of Feat's displays for analysis presentation
3. the use of additional displays built outside of Feat for results presentation

6.3 Future Directions

ERF is baselined for use in the control center, so where do we go from here? We intend to explore alternate analysis engines and representations, communication with other applications, development of more sophisticated FDI algorithms, and the recovery problem. Some of these efforts have already started, while others are still on the horizon.

7. CONCLUSION

ERF is an application to aid a Mission Controller (MC) in identifying the cause(s) and subsequent effects of observed failure symptoms in a monitored system. It is a layer of software built upon the Failure Environment Analysis Tool (FEAT) provided by JSC's Intelligent Systems Branch. The additions that the "ER" in ERF bring are; 1) automated fault identification and effects analysis algorithms which are model independent, 2) hooks for alternate model representations and alternate analysis engines, 3) interfaces to real-time data, and 4) automated problem management functions.

ERF uses advanced automation techniques. It provides automated Fault Detection and Identification (FDI) analysis for any system which can be modeled

as a causal network of failure modes. As the MC identifies additional candidate functions for automation, there will be opportunity for ERF's capabilities to expand and/or for it to work with other applications. ERF has been designed to provide hooks for swapping underlying representations and analysis engines, for incorporation of more advanced analysis algorithms, and for communication with other applications.

ERF is a technology transfer project, having moved from the labs into the operational world. Its requirements are derived from a subset of the MC's FDIR task description and implemented via advanced automation techniques. It has MC support, having been designed not only with the end-user in mind but with the end-user having actively participated in the design process. Hence, its design has been influenced by an understanding of the operational environment, the MC's FDI task, and the capabilities of automation technology.

REFERENCES

1. *Functional Description* (Internal Document), April 2, 1991.
2. *Space Station Control Center User Detailed Functional Requirements*. April 1991. JSC-13192.
3. Malin, J. T., and Schreckenghost, D. L. 1992. *Making Intelligent Systems Team Players: Overview for Designers*. NASA Technical Memorandum 104751.
4. *Macintosh FEAT 3.4 User's Guide*. 1992. JSC's Intelligent Systems Branch. August 24, 1992. (includes the Digraph Editor User's Guide)
5. *Unix FEAT 3.4 User's Guide*. 1992. JSC's Intelligent Systems Branch. August 24, 1992.
6. Stevenson, R. et al. 1991, Failure Environment Analysis (FEAT) Tool Development Status. Presented at *AIAA Computing And Aerospace VIII Conference*. Baltimore, MD: AIAA 91-3803.
7. Clark, Colin et al. 1992. Fault Management For The Space Station Freedom Control Center. Presented at *AIAA 30th Aerospace Sciences Meeting & Exhibit*. Reno, NV: AIAA 92-0870.
8. *Space Station Control Center Extended Real-Time FEAT Subsystem Specification (Preliminary Draft)*. March 1992. JSC-13416.

FEAT	ERF
Simple analysis -limited to single queries	Automated analysis using sequenced FEAT calls: Commonality Assessment -quick look commonality assessment of multiple out-of-limit and alarm conditions Find Cause -uses information about what has and has not failed Predict Effects (built using the Find Cause results) Immediate consequences Lost redundancies New system susceptibilities (Next worst failure)
Analysis uses only failure mode information (i.e., does not use what is known to be good)	Automated Analysis using both failure and non failure information
No real-time problem management capabilities	Real-time problem management functions
Manual entry of system configuration data to reflect degraded system conditions	Semi-automated entry of system configuration information for tracking degraded system conditions (e.g., The Mission Controller records system degradation via some TBD electronic method. ERF will use this information to automatically configure the system model to reflect the system's degraded condition.).
Manual entry of annunciated alarms and out-of-limit conditions	Automated entry of annunciated problem conditions
Manual retrieval of additional status information not contained in the problem announcement	Automated retrieval of additional information needed for automated analysis
	Automated entry (upon MC direction) of telemetry data reflecting current conditions when no problem has been announced
No distinction between "real" and "hypothetical" analysis	"Real" vs "Hypothetical" problem management: "Real" mode for use of real telemetry information "Hypothetical" mode for "what if" analysis Initialization of "hypothetical" mode using actual, instant conditions Provision to update "real" information from selected "hypothetical" data (note: such updates do not automatically propagate to other systems)
No hooks readily available for additional analysis algorithms, alternate on-line analysis engines, alternate model representations, etc.	Especially designed to provide hooks for: Alternate/additional analysis algorithms (well beyond transitive closure) Alternate/additional engines for performing analysis Alternate model representations (e.g., expansion to full propositional logic) Communication with other analysis applications

THE MISSION EVENTS GRAPHIC GENERATOR SOFTWARE: A SMALL TOOL WITH BIG RESULTS

N94-23893

Mark Lupisella, Jack Leibee, Charles Scaffidi

NASA/Goddard Space Flight Center
Greenbelt, Maryland USA

ABSTRACT

Utilization of graphics has long been a useful methodology for many aspects of spacecraft operations. This paper presents a personal computer based software tool that implements straight-forward graphics and greatly enhances spacecraft operations. This unique software tool is the Mission Events Graphic Generator (MEGG) software which is used in support of the Hubble Space Telescope (HST) Project. MEGG reads the HST mission schedule and generates a graphical timeline.

Key Words: mission operations, mission schedule, graphics, automation

1. INTRODUCTION

The general purpose of this paper is to introduce the MEGG software at a conceptual level. If additional technical information is desired, the author would be happy to provide it.

Section 2 presents an overview of the HST Ground System and the MEGG software. Section 3 discusses the operational needs and design features incorporated in MEGG. Section 4 addresses lessons learned. Section 5 discusses future plans for the software. Section 6 summarizes. Section 7 provides references.

2. OVERVIEW

2.1 HST Ground System Overview

This paper begins with a high level introduction to the HST Ground System in order to provide a general context in which the MEGG software role can be understood.

The HST Ground System consists of science systems and control center systems. The science systems are located at the Space Telescope

Science Institute (ST ScI) in Baltimore, Maryland. The control center systems are located at the Goddard Space Flight Center in Greenbelt, Maryland. Figure 2.1 shows a high level functional diagram of these systems. The primary process begins with a proposal being submitted by the scientific community to the ST ScI. Once approved, it is incorporated into a Science Mission Schedule (SMS), generated by the ST ScI, which is a detailed science schedule spanning one week.

The SMS is transmitted to the control center where it is processed with orbit data and Tracking and Data Relay Satellite System (TDRSS) schedule data to generate the mission schedule, TDRSS file, and command loads for one week. The command loads are sent to the on-line operations system and then uplinked to the spacecraft via the NASA Communications (NASCOM) Space Network.

On the return link, the spacecraft sends science and engineering telemetry to the on-line operations system. Astrometry and engineering data is processed and sent to the ST ScI. The spacecraft also returns 1 megabit science data directly to a dedicated data capture facility where it is then sent to the ST ScI. The ST ScI processes and archives the data for use by the scientific community.

2.2 MEGG Overview

2.2.1 Operational Function

MEGG's primary operational function is to translate the HST mission schedule file and the TDRSS schedule file into a graphic timeline representing the spacecraft activities for one week. Both files are ASCII reports. In general, the mission schedule file is 8,000 pages long and takes 8 megabytes of disk space. The TDRSS file is usually 1.2 megabytes in size. MEGG is an

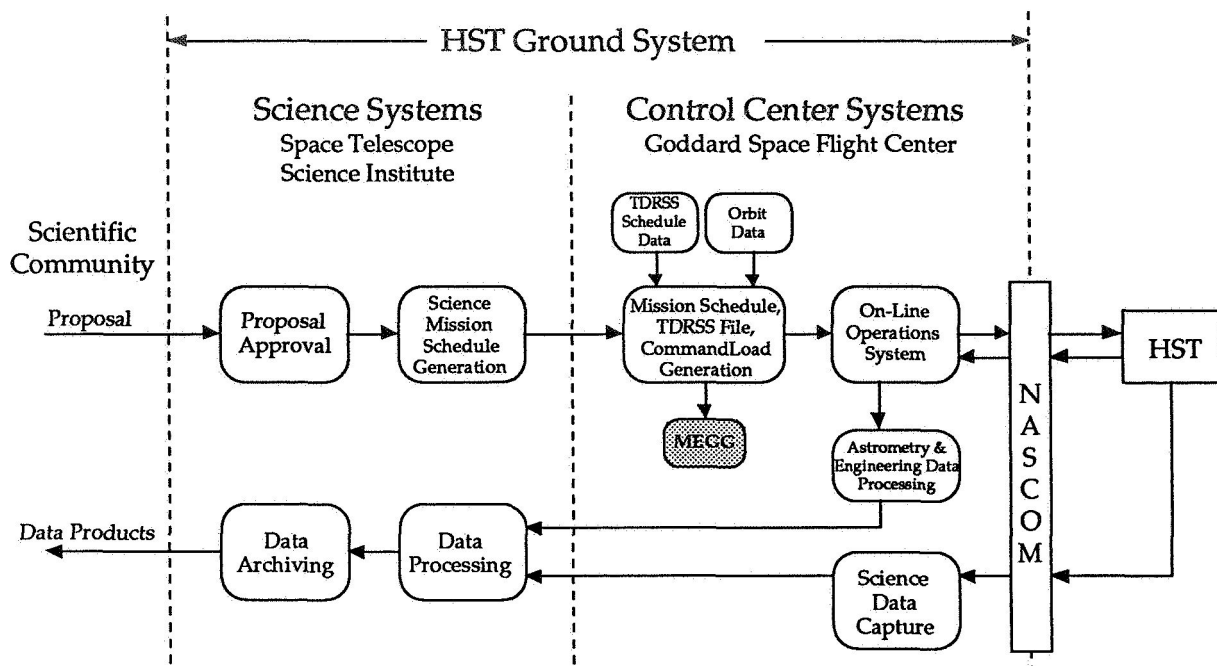


Figure 2.1: HST Ground System Functional Overview

automated system that generates a product used by the full spectrum of individuals ranging from operations engineers to scientists. Because of the sophisticated and complicated nature of HST, the need for this kind of tool emerged early in the pre-launch testing phase. Initially, the need was so great, the graphic timeline was generated manually. However, it became imperative to automate this system and increase its operational functionality. Computer Technology Associates (CTA) originally developed the software and it has since been enhanced and maintained as a NASA in-house software product.

2.2.2 Hardware and External Interfaces

Figure 2.2 shows the MEGG hardware components and its interface to the HST Ground System. As mentioned, the ST Scl generates the SMS and transmits it directly to one of the HST control center mainframe computers. The Flight Dynamics Facility sends HST orbit data to this mainframe and the Network Control Center, which is responsible for scheduling and operations of the TDRSS satellites, transmits the TDRSS schedule data to the mainframe. Here, command files, the mission schedule, and the

TDRSS file are generated by the HST off-line support software. The command files and TDRSS file are used to generate the command load which is sent to the on-line operations system. The mission schedule and TDRSS file are sent to the MEGG system via an ethernet connection. With the mission schedule and TDRSS file both residing on the personal computer, MEGG is ready to be executed and produce a PostScript file which the printer converts into graphics. The final 85 page hard copy of the timeline can then be distributed accordingly.

2.2.3 Software Overview

MEGG uses three programming languages: C, AWK, and PostScript. The user interface and controlling code module is written in C. The user interface is menu driven. This module also controls the flow of the software during a run.

AWK is a pattern recognition language which closely resembles C but has its own interpreter. The code developed in AWK does most of the processing during a MEGG run. The modules written in AWK read the input files (the mission schedule file and TDRSS file) and create the

event list of recognized activities in these input files. The main modules of code written in AWK are the Spacecraft Support Module (SSM), the Science Instrument Module (SIM), and the TDRSS File Module (TFM). The SSM and SIM both read the mission schedule separately and generate their corresponding events for the event list. The TFM reads the TDRSS file and generates events for the same event list.

Additional AWK modules, along with the Postscript kernel module and graphic-defining module, also produces the final Postscript file which is sent to a laser printer. PostScript is an advanced graphics language used by some laser printers to produce sophisticated graphics.

3. OPERATIONAL NEEDS AND DESIGN FEATURES

3.1 Automation

As mentioned previously, graphic representations of mission schedules were generated manually during the pre-launch testing phase. It was soon recognized that the

process needed to be automated in order to be incorporated as an efficient operational system.

The primary design features that accomplish automation are automatic input of ASCII report files, mentioned in Section 2.2.1, and the use of different programming languages for different tasks. The main tasks performed by MEGG are pattern recognition and graphics generation. The graphical user interface is written in C.

Pattern recognition is accomplished using the programming language, AWK. It closely resembles C but is easier to use for pattern recognition. AWK was designed primarily for high speed pattern recognition.

Graphics are generated using the PostScript graphics language. PostScript is versatile and compatible with most printers and allows for detailed graphic representation.

3.2 Wide Range of Applicability

A tool was needed that could be used by the full spectrum of individuals and serve a wide

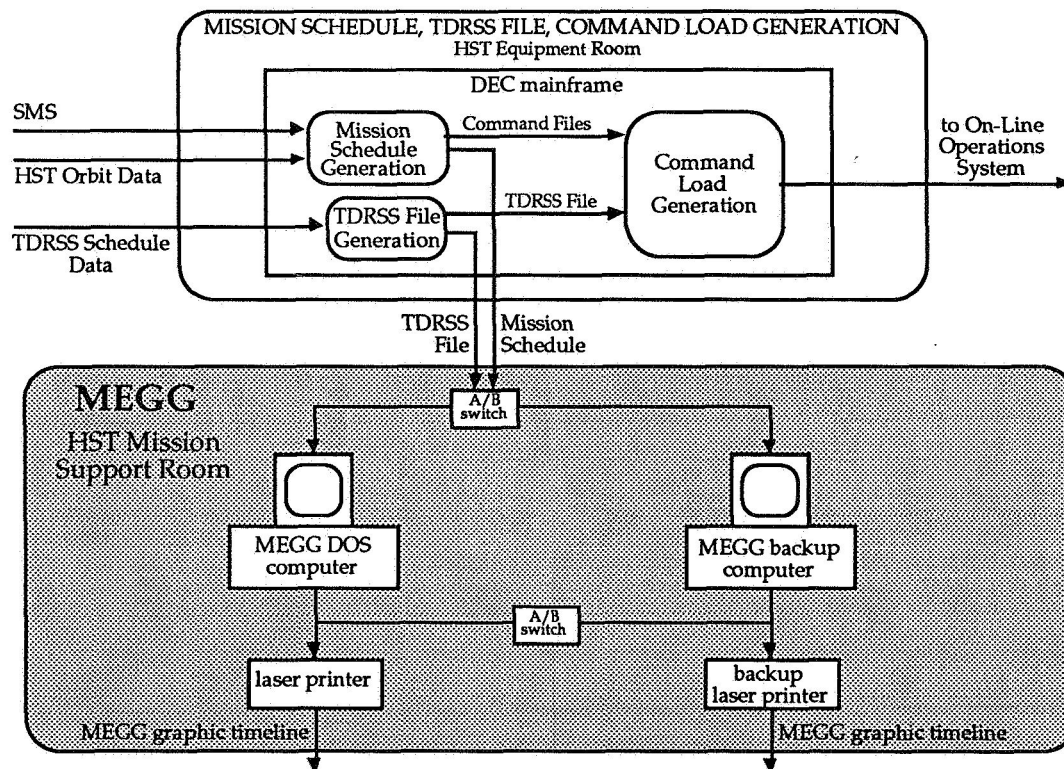


Figure 2.2: MEGG Hardware/Interface Overview

range of functionality. Scientists, engineers, console operators, and managers have to be able to reference the same source of schedule information in a simple and efficient manner. MEGG serves this purpose. It is used for schedule validation and review, real-time operations, and off-line analysis. The design features used to satisfy the need for a wide range of applicability are comprehensive and tailored graphics.

The MEGG timeline is comprehensive because it displays information about all the major elements of the spacecraft's operation. It has functionality for all users. Figure 3.1, the MEGG key, gives an idea of what the timeline looks like and shows the range of information covered by the timeline. A more typical page would have less information and would show correlations between different lines of the timeline.

In addition, the timeline's graphics are tailored to meet specific functional needs of people interested in the mission schedule. Again, this can be seen in Figure 3.1. This allows for schedule review of specific events, visual access to the schedule for console engineers, and off-line analysis for specific events and time periods. Efficient schedule review is facilitated by grouping related information together to provide the most efficient visual representation. In addition, MEGG has often alerted engineers to potential difficulties in the schedule. These difficulties can then be resolved before they become a problem for on-line operations. Using the MEGG output, console engineers are able to more closely follow operations events associated with the spacecraft, resulting in a better overall awareness. MEGG also serves as a valuable off-line analysis tool since it can be referenced to understand what specific events were happening at specific times such as during anomalies.

3.3 Simple Interactive Use

It was desirable to design software that was not difficult to learn or use. This allows for efficient operations and training.

The user interface was designed to be straightforward and easy to use. It is menu driven and has on-line help. If the operator makes an error, notification and an explanation is provided.

Options for execution of the software are limited to reduce possibility of operator error.

Also, MEGG is personal computer based and therefore is inherently easier to use and understand compared to the usual mainframe computer or workstation.

3.4 Ease of Enhancement and Maintenance

It was desirable to have a software tool that was very easy to enhance and maintain. This would allow for timely deliveries when needed.

The primary design feature that allows for easy enhancement and maintenance is the use of modular coding. There are eight modules of developed code in the MEGG software. Conceptually, this makes the code easy to understand and hence modify. Also, modular coding allows for efficient troubleshooting since problem areas can easily be isolated.

Again, the fact that the software runs on a personal computer makes it easy to maintain and develop enhancements. A personal computer can be used as an isolated system, thereby relieving resource conflicts. Also, personal computers are relatively inexpensive making transfer of the software to other groups less of a financial impact.

4. LESSONS LEARNED

4.1 Replace AWK With C

It is likely that a better choice of language for the pattern recognition processing is C. While AWK specializes in high speed pattern recognition, and was a good choice for that reason, C could do the same job and offers more versatility and easier portability. It is a language that most programmers are familiar with. In addition, using C would alleviate the need for using and obtaining AWK interpreters for different operating systems. This is important when considering implementation into other systems. The cost effectiveness and performance impacts of this change require further investigation.

4.2 Perform One Read Pass Of Mission Schedule

MEGG presently takes two passes to read the mission schedule. The first pass reads for

engineering (health and safety) events, and the second pass reads for science instrument events. This is useful for development and conceptual understanding of the software, however, it is CPU intensive. This could be reduced if only one pass was executed to read for all events. This would be an easy modification and would reduce the complexity of the software as well as the overall run-time.

5. FUTURE PLANS

5.1 Implement Lessons Learned

The AWK subroutines will be re-coded in C for increased versatility and portability. In addition, the code will be restructured to make only one read pass of the mission schedule.

5.2 Servicing Mission

In late 1993, NASA will launch a mission to service the HST. A new Wide Field Planetary Camera (WFPC II) and the corrective optics (COSTAR) will be installed. Additional servicing tasks will be executed but only the above mentioned affect the MEGG software.

The MEGG software needs to be modified to incorporate the two new instruments. New command structures for WFPC II and relevant operational states for COSTAR need to be investigated and implemented so that all relevant information is accurately represented.

5.3 Incorporate MEGG Into Other Systems

The system wide uses for the MEGG software are numerous. The HST Flight Software group has expressed interest in implementing the software into their system. Also, MEGG would be very useful in the HST trend analysis system since engineering information could readily be referenced against a comprehensive visual schedule of spacecraft events. The Telemetry Analysis for Operation Support (TALOS), a supplement system to the core operations software, has incorporated the MEGG display into its video graphics.

Ultimately, it is desirable to incorporate MEGG into the core operational software allowing console engineers instant access to a chosen aspect of the graphical representation of the

mission schedule. This could ultimately lead to artificial intelligent based systems which alert operators to various events and required procedures.

6. SUMMARY

MEGG is a software tool that greatly enhances HST operations by efficiently and graphically representing the mission schedule. It is a small tool that provides an important function.

The operational needs of automation, wide range of applicability, simple interactive use, and easy maintenance are satisfied by hardware components and software design.

Future plans for MEGG involve implementing lessons learned, preparing for the HST servicing mission, and incorporation into other systems.

7. REFERENCES

Computer Technology Associates, (1988), *Hubble Space Telescope Mission Events Graphic Generator Final Report*, prepared for NASA/Goddard Space Flight Center

Computer Technology Associates, (1988), *Hubble Space Telescope Mission Events Graphic Generator Developer's Guide*, prepared for NASA/Goddard Space Flight Center

Computer Technology Associates, (1988), *Hubble Space Telescope Mission Events Graphic Generator Sourcecode Handbook*, prepared for NASA/Goddard Space Flight Center

Lupisella, M. (1992), *Hubble Space Telescope Mission Events Graphic Generator Requirements Document*, NASA/GSFC Mission Operations Division, Control Center Systems Branch

NASA/Goddard Space Flight Center, Mission Operations Flight Software Systems Branch and Control Center Systems Branch, (1992), *Hubble Space Telescope Mission Events Graphic Generator User's Guide*

SESSION F

OPERATIONS AUTOMATION AND EMERGING TECHNOLOGIES

CO-CHAIRPERSONS

G. A. Giffin, NASA, USA

D. N. Sweetnam, JPL, California Institute of Technology, USA

Summary	401
D. N. Sweetnam	

F.1 Influence of a New Generation of Operations Support Systems on Current Spacecraft Operations Philosophy: The Users Feedback	403
J.-M. Darroy, Matra Marconi Space, Toulouse, France	

F.2 PILOT: An Intelligent Distributed Operations Support System	411
A. N. Rasmussen, The MITRE Corporation, Houston, Texas, USA	

F.3 ESSOPE: Towards Spacecraft Operations with Reactive Schedule Planning	417
J. Wheadon, ESA European Space Operations Centre, Darmstadt, Germany	

F.4 Verification and Validation of Expert Systems Used in Ground Data Systems	423
C. Culbert, NASA Johnson Space Center, Houston, Texas, USA	

F.5 The Advanced Technology Operations System ATOS	425
J.-F. Kaufeler and H. A. Laue, ESA European Space Operations Centre, Darmstadt, Germany; K. Poulter, Logica Cambridge Limited, London, UK; H. Smith, Logica Space and Communications Ltd., London, UK	

F.6 NASDA Knowledge-Based Network Planning System	435
K. Yamaya, M. Fujiwara, and S. Kosugi, NASDA, Tokyo, Japan; M. Yambe, Fujitsu Ltd., Tokyo, Japan; M. Ohmori, Fujitsu Social Systems Engineering Ltd., Tokyo, Japan	

F.7 A Cross-Disciplinary Response to Improve Test Activities: The Corporate Memory Capitalization in Ariane4 Test Domain	441
D.-P. Vo and C. Soler, Aramiihs-Matra Marconi Space, Toulouse, France; N. Aussenac and D. Macchion, Aramiihs-Institut de Recherche en Informatique de Toulouse, France	

F.8 NASA's Artificial Intelligence Research Project: Lessons Learned	451
M. D. Montemerlo, NASA, Washington, D.C., USA	

F.9 Combining Real-Time Monitoring and Knowledge-Based Analysis in MARVEL	453
U. M. Schwuttke, A. G. Quan, R. Angelino, and J. R. Veregge, JPL, California Institute of Technology, Pasadena, California, USA	

F.10 The Application of Automated Operations at the Institutional Processing Center	459
T. H. Barr, OAO Corporation, Altadena, California, USA	
F.11 Making Tomorrow's Mistakes Today: Evolutionary Prototyping for Risk Reduction and Shorter Development Time	465
G. Friedman, U. Schwuttke, S. Burliegh, S. Chow, R. Parlier, L. Lee, H. Castro, and J. Gersbach, JPL, California Institute of Technology, Pasadena, California, USA	
F.12 NASDA's Advanced Online System (ADOLIS)	471
Y. Yamamoto, H. Hara, and S. Yamada, NASDA, Tokyo, Japan; N. Hirata, S. Komatsu, S. Nishihata, A. Oniyama, NEC Corporation, Yokohama, Japan	
F.13 Constraint-Based Scheduling	477
M. Zweben, NASA Ames Research Center, Moffett Field, California, USA	
F.14 Customizing Graphical User Interface Technology for Spacecraft Control Centers	485
E. Beach and P. Giancola, Computer Sciences Corporation, Laurel, Maryland, USA; S. Gibson, Integral Systems, Inc., Lanham, Maryland, USA; R. Mahmot, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA	

SUMMARY — SESSION F

D. N. SWEETNAM
Jet Propulsion Laboratory

The most positive component of this session was the breadth of participation: four NASA centers, Germany, France, the United Kingdom, and Japan were represented. Geoff Giffin (Deputy Director, Space Science and Operations Division, OAST, NASA Headquarters) gave the introduction, in which he stressed that we are entering an era of limited budgets wherein new technologies and consequent automation will play a stronger role in maintaining our operations capabilities. A list of presenters and summaries of the papers are given below:

F.1, "Influence of a New Generation of Operations Support Systems on Current Spacecraft Operations Philosophy: The Users Feedback," J.-M. Darroy. The author discussed several expert systems, including Plan ERS for automated mission planning and scheduling; POM for automated operations procedures; Arianexpert for postflight mission monitoring and performance analysis; and Telecom 2 expert for automated failure diagnosis. One significant user input was "Deliver ASAP."

F.2, "PILOT: An Intelligent Distributed Operations Support System," A. N. Rasmussen. The author discussed work with PILOT, which is a facility for managing the NASA Johnson Space Center's real-time telemetry system called RTDS. He indicated that PILOT will integrate the individual workstations and that use of expert shells such as G2 is being examined. PILOT is currently an advisory-only application but is being expanded to allow hands-off mission operations.

F.3, "ESSOPE: Towards Spacecraft Operations with Reactive Schedule Planning," J.-F. Kaufeler. The Expert System for Spacecraft Operations Planning and Execution (ESSOPE) is used with the Olympus geostationary satellite. Performance is such that a 24-hour schedule is processed in 2 to 3 minutes.

F.4, "Verification and Validation of Expert Systems Used in Ground Data Systems," C. Culbert. The author discussed the use of verification and validation (V&V) methods with expert systems as a mechanism for increasing their acceptance. Guidelines for KBS developers were

presented; the use of surveys and workshops for V&V training were described as effective methods.

F.5, "The Advanced Technology Operations System ATOS," H. Smith. The presenter described a program used at the ESA European Space Operations Centre to put in place standards to support machine-human knowledge bases that will be used with ATOS. He discussed an operations ontology, development of a formal language to specify the ontology, knowledge sharing between missions and its reuse, and the idea that "knowledge is the corporate asset." ATOS advanced application modules for operations preparation, mission planning, operations execution, and training were also discussed.

F.6, "NASDA Knowledge-Based Network Planning System," M. Ohmori. The presenter discussed the architecture of a planned network-planning system and several methods for conflict resolution in resource planning, including cut and try, priority, and rejection. He indicated that the planner is expected to be useful for simulation. Performance will be able to handle 28 satellites and 12 antennas; a monthly schedule is processed in two hours.

F.7, "A Cross-Disciplinary Response to Improve Test Activities: The Corporate Memory Capitalization in Ariane4 Test Domain," D.-P. Vo. The presenter discussed the need for corporate memory in the Ariane4 High Equipment Bay in the control of equipment that ensures mating between the third stage and the satellite payload. Components of fault diagnosis were also discussed, including historical databases of design information as well as old incidents. A system to capture the historical knowledge and develop an "assistant" is underway.

F.8, "NASA's Artificial Intelligence Research Project: Lessons Learned," G. A. Giffin. Operations at NASA are beginning to reap the benefits of the AI program. RTDA at NASA Johnson Space Center is an example, and may have a larger future payoff in that there may only be a single operations center for control of both the Shuttle and

Space Station Freedom. Payback on investment needs to be three years in order to get funded. A major lesson: find a user on the inside and learn what they have to say and what they need.

F.9, "Combining Real-Time Monitoring and Knowledge-Based Analysis in MARVEL," U. M. Schwuttke. MARVEL's event-driven activation of reasoning monitors spacecraft subsystems such as command and control, attitude control, flight data, and telecommunications. A future need is a system-level knowledge base. Current implementation supports Voyager; future implementation will support the Galileo mission. Voyager use has demonstrated the potential for major (up to 60 percent) reductions in the downlink spacecraft team staffing.

F.10, "The Application of Automated Operations at the Institutional Processing Center," T. H. Barr. Various methods of automating activities at IPC were discussed, including commercial tools as well as "in-house" developments such as Keymaster, a PC-based support system, Docutext, and a semi-automated failure reporting system.

F.11, "Making Tomorrow's Mistakes Today: Evolutionary Prototyping for Risk Reduction and Shorter Development Time," G. Friedman. The presenter described the accomplishments of the Flight Projects Office Information Systems Testbed (FIST), such as VNESSA and SANTA2. Topics included the advantages of rapid prototyping, such as risk reduction, reduced development cycle, accelerated technology transfer, and better user interaction. Future activities, such as the Small Mission Prototype, were also discussed.

F.12, "NASDA's Advanced Online System (ADOLIS)," S. Komatsu. The presenter discussed the real-time mission operations control system, which is ultimately intended for manned mission support, and its multimedia features, including a camera and video display, speaker/microphone, image scanner, and tablet. Additional features include a bilingual touch switch from Japanese to English. Future capabilities should include voice translation.

F.13, "Constraint-Based Scheduling," M. Zweben. The author discussed a scheduling tool (GERRY) used during Shuttle refurbishment between launches in the Orbiter Processing Facility at NASA Kennedy Space Center. The tool includes scheduling of inspection and repair activities for engines, tiles, and payloads. The technique uses hard rules and preference criteria and a mechanism for optimization. Other applications include scheduling of wind tunnel resources at NASA Ames Research Center.

F.14, "Customizing Graphical User Interface Technology for Spacecraft Control Centers," S. Gibson. The presenter described the graphical user interface for TPOCC, the real-time mission control system at NASA Goddard Space Flight Center. The interface uses HP-VU, with workspaces in Rooms, and OSF Widgets, such as text boxes, scrolling text windows, graphical display windows, and composite windows.

N94-23894

INFLUENCE OF A NEW GENERATION OF
OPERATIONS
SUPPORT SYSTEMS
ON
CURRENT SPACECRAFT OPERATIONS
PHILOSOPHY :

THE USERS FEEDBACK

Jean Michel DARROY

tel. : (33) 61.39.67.35

Fax : (33) 62.24.77.80

MATRA MARCONI SPACE

31 rue des Cosmonautes
Z.I du Palays
31077 TOULOUSE CEDEX
FRANCE

ABSTRACT

Current trends in the spacecraft mission operations area (spacecraft & mission complexity, project duration, required flexibility,...) are requiring a breakthrough for what concerns philosophy, organization and support tools.

A major evolution is related to space operations "informationalization" (ref 1), i.e adding to existing operations support & data processing systems a new generation of tools based on advanced information technologies (object-oriented Programming, artificial intelligence, data bases, hypertext,...) that automate, at least partially, operations tasks that used be performed manually (mission & project planning/scheduling, operations procedures elaboration & execution, data analysis & failure diagnosis,...)

All the major facets of this "informationalization" are addressed at MATRA MARCONI SPACE, operational applications have been fielded and generic products are becoming available.

These various applications have generated a significant feedback from the users (at ESA, CNES, ARIANESPACE, MATRA MARCONI SPACE,...), which is now allowing us to precisely measure how the deployment of this new generation of tools, that we called OPSWARETM, can "reengineer" (ref 2) current spacecraft mission operations philosophy, how it can make space operations faster, better and cheaper.

This paper can be considered as an update of the keynote address "Knowledge-Based Systems for Spacecraft Control" (ref3) presented during the first "Ground Data Systems for Spacecraft Control" conference in Darmstadt, June 1990, with a special emphasis on these last two years users feedback.

KEYWORDS : Mission control, operations, automation, artificial intelligence, object-oriented programming, planning, scheduling, procedures, diagnosis, documentation, reengineering.

1 - INTRODUCTION

A major lesson of the 1st "Ground Data Systems for Spacecraft control " conference in Darmstadt, June 1990, was the emergence of new operations support tools, based on advanced information technologies (e.g artificial intelligence), in various domains such as mission planning, operators assistance or failure diagnosis (ref 3).

The deployment of these first tools was very promising, but appeared at that time as separated initiatives. Things have changed at lot during the past two years, and a more global and more significant evolution can be noticed today, affecting the nature and also the economics of space operations.

2 - NEW CHALLENGES FOR SPACE OPERATIONS

Current trends in spacecraft mission operations are requiring a breakthrough for what concerns philosophy, organization and support tools, in order to meet a sufficient level of safety, productivity and mission return :

- a - the human operator is facing spacecraft which are becoming more and more complex
- b - space missions management complexity is increasing
- c - space projects duration is continuously increasing, thus creating problems of expertise & experience capture
- d - space systems require more and more flexibility and adaptability
- e - space missions are generating huge amount of data, from which essential informations are very difficult to extract.

Such a breakthrough is currently being implemented by the "informationalization" (ref1) of space operations, i.e a more efficient management of informations generated and used by space operations, and the automation of operations tasks, which used to be performed manually. This "informationalization" corresponds to the deployment of a complete set of tools, in addition to current operations and data processing systems. We propose to call OPSWARE™ this new level in ground data systems. OPSWARE™ covers all the major mission operations tasks, as presented at figure 1.

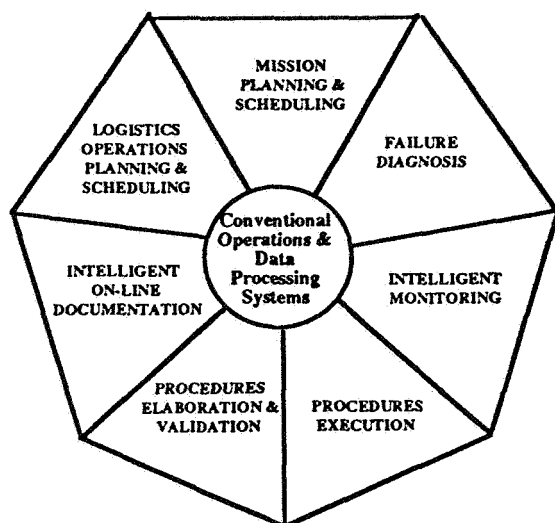


Fig. 1 - OPSWARE™ :
A complete set of tools

The major evolution , since the previous conference in Darmstadt, has been the global and consistent deployment of this set of tools, while two years ago, only isolated applications were paving the way for this larger wave.

And clearly, some applications, which were developed during this period, have demonstrated significant results with reference to safety, productivity or mission return .

3 - SOME BRILLIANT ILLUSTRATIONS

During the past two years, numerous OPSWARE™ applications have been deployed, which address the various facets of OPSWARE™ as presented at figure 1.

Figure 2 gives a non-exhaustive list of some examples, developed by MATRA MARCONI SPACE .

OPSWARE™ FACET	CUSTOMER(S)	APPLICATION
MISSION PLANNING & SCHEDULING	ESA/ESTEC	PLAN ERS FOR ERS-1 EARTH OBSERVATION SATELLITE MISSION STRATEGY DEFINITION AND SIMULATION
LOGISTICS OPERATIONS PLANNING & SCHEDULING	MD4S	ARIANE 4 EQUIPMENT BAY ASSEMBLY, INTEGRATION & VALIDATION
INTELLIGENT ON-LINE DOCUMENTATION	MD4S	COMMUNICATIONS SATELLITES OPERATIONS MANUAL
PROCEDURES ELABORATION & VALIDATION	CNES, MD4S, ESA	POM FOR TELECOM2, HISPASAT & SOHO PROCEDURES ELABORATION
	ESA/ESTEC	PREVISE FOR MANNED FLIGHT PROCEDURES ELABORATION & VALIDATION (SPACELAB, COLUMBUS)
	CNES	PROCSAT FOR SPORT EARTH OBSERVATION SATELLITE PROCEDURES ELABORATION & VALIDATION
PROCEDURES EXECUTION	ESA/ESOC	EXPERT OPERATOR ASSOCIATE FOR MARECS B2 SPACECRAFT OPERATORS ASSISTANCE
	ESA/CNES	CREW SUPPORT SYSTEM FOR ASTRONAUTS ASSISTANCE & TRAINING
INTELLIGENT MONITORING & DATA ANALYSIS	ARIANESPACE	ARIANEXPERT FOR ARIANE 4 POST-FLIGHT DATA ANALYSIS
	MD4S	TELECOM2 & HISPASAT SATELLITES PERFORMANCES ANALYSIS
FAILURE DIAGNOSIS	CNES	TELECOM2 FAILURE DIAGNOSIS

Fig.2 OPSWARE™ APPLICATIONS
DEVELOPPED BY MATRA MARCONI
SPACE

It is very interesting to notice that most of these applications have produced significant results. Let's have a look at some brilliant illustrations.

Plan ERS was used by ESA/ESTEC engineers in 1991 for testing various mission strategies for ERS-1 earth observation satellite (ref 4). The main features of the system were :

- a very flexible planning system thanks to an object-oriented representation of the domain (satellite, resources,...) and a rule-based representation of the mission strategies (e.g. on-board recorder management).
- a powerful planning system, which can handle more than 4000 request in less than 30 minutes

- an environment for mission strategies definition and update, thanks mainly to a syntax-driven editor.

The feedback we got from the users was very positive . Plan ERS allowed the identification of the appropriate mission strategy for optimizing mission products, and thus customer's satisfaction.

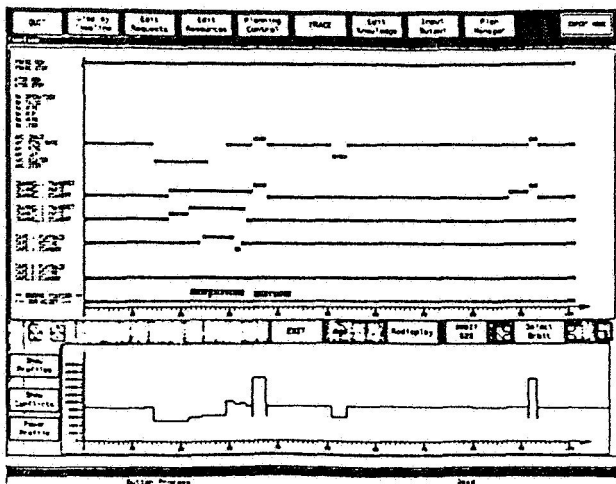


Fig. 3 PLAN ERS TIMELINES & RESOURCES CONSUMPTION PROFILES

POM is a tool developed by MATRA MARCONI SPACE for operations procedures elaboration (ref 5). It has been used in 1991 and 1992 on various satellites projects : TELECOM2, HISPASAT, SOHO.

POM is based on a formal representation of procedures, and on a customized procedure editor. The main benefit of using this tool on the here-above mentioned projects was a significant reduction of operations preparation phase duration (50%).

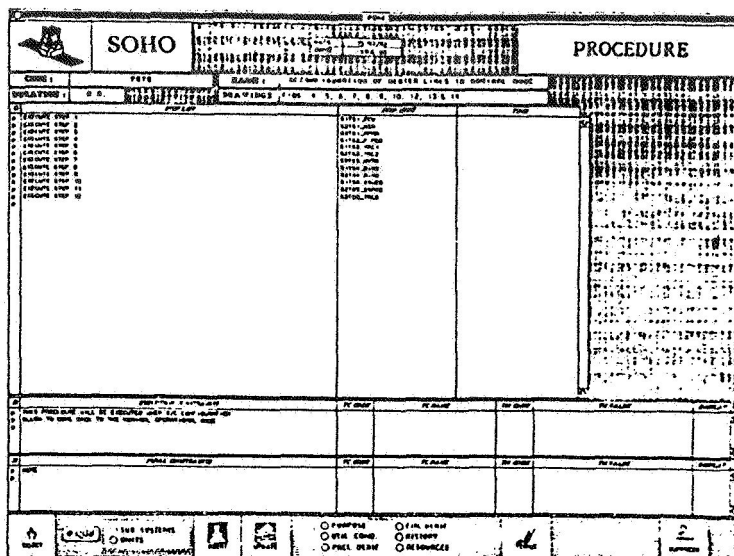


Fig. 4 - POM PROCEDURE EDITOR

ARIANEXPERT has been operationnally used at ARIANESPACE since january 1991 for ARIANE 4 launcher post-flight analysis (ref 6). Its main features are :

- a rich set of signal processing and graphical tools
- a procedure-based analysis definition
- a tree-based procedures conditional chaining
- flexible syntax-driven and graphical editors
- the generation of a technical memory (archiving of all analysis results)
- automatic report formating through a word processor

The measured benefits of ARIANEXPERT use are significant : the analysis duration is reduced by 75 % while the analysis is more detailed and goes further than previously and the analysis is more exhaustive and more systematic.

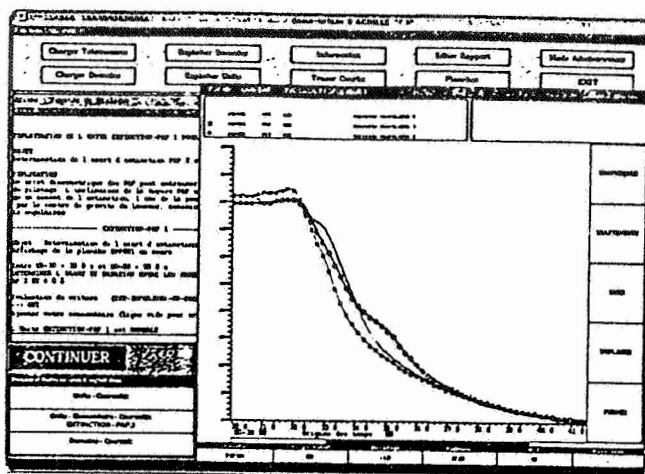


Fig. 5 - ARIANEXPERT DATA VISUALIZATION

TELECOM 2 EXPERT is a knowledge-based failure diagnosis system developed for CNES operators (ref.7). A first complete version of the system has been available since June 1992, and the system will be formally delivered to CNES in January 1993. It is a powerful failure diagnosis system based on various types of knowledge : decision trees encoding the global spacecraft behaviour, and a collection of subsystem functional models capturing the detailed links and interactions that exist between the various spacecraft components. This system is also considered by CNES as an efficient training tool, thanks to its deep knowledge of the spacecraft and thanks to the possibility of session replay. It is also viewed as generating a mission technical memory, allowing the integration of in-orbit experience. Its main benefits are related to expertise transfer from spacecraft designers to spacecraft operators, and to in-orbit experience capture (a critical point, if we consider the important turnover in operations staff).

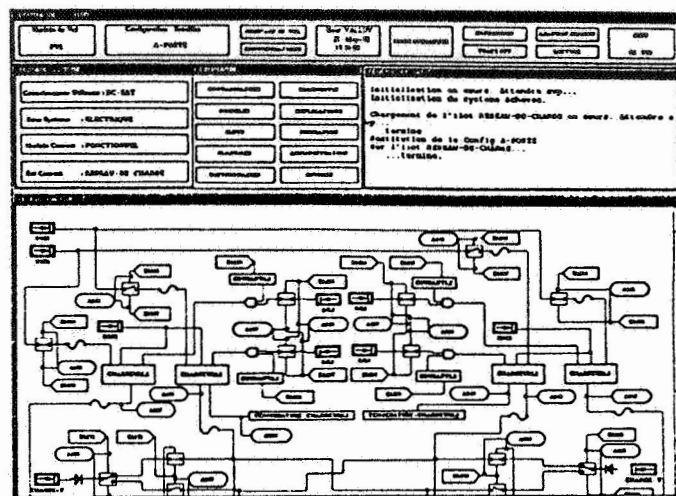


Fig. 6 - TELECOM 2 EXPERT PROPOSED ACCESS TO SPACECRAFT MODELS

Other brilliant OPSWARE™ applications could be mentioned, developed either by MATRA MARCONI SPACE or by other space organizations or companies. These ones clearly demonstrate the deep influence OPSWARE™ tools already have on space operations quality, safety and productivity.

It is also of prime importance to consider that all these tools are no more developed as completely isolated systems (even if such or such particular project context may lead to this type of conclusion), but are elementary building blocks of a more general concept, the so-called OPSWARE™ level. That is clearly the approach which has been adopted at MATRA MARCONI SPACE (Ref 8), and also by other organization such as ESA/ESOC with the ATOS programme (ref.9).

4 - FROM CUSTOMIZED APPLICATIONS TO GENERIC PRODUCTS

Another interesting evolution is the emergence of generic products for developing OPSWARE™ applications. Such an evolution definitely shows the maturity OPSWARE™ domain is gaining.

Four generic OPSWARE™ products, developed by MATRA MARCONI SPACE, pretty well illustrate this evolution.

X-ANALYST is a generic product for intelligent data analysis, which has been derived from the ARIANEXPERT application (see § 3). All the technical features which were mentioned for ARIANEXPERT are present within this tool, but they have been made more generic. Since then, X-ANALYST has been applied to other contexts, and more specifically satellites trends analysis. These applications are used by MATRA MARCONI operations department for TELECOM2, and HISPASAT communications satellites, in the context of its customers' support activities. Here again, the main benefits are the reduction of analysis duration and the analysis quality. X-ANALYST application to SPOT2 and SPOT3 satellites is planned.

DIAMS is a generic product for spacecraft failure diagnosis, which is derived from the TELECOM2 EXPERT application (see § 3). Knowledge representation languages, diagnosis mechanisms, development environment and man-machine interface, which exist in TELECOM2 EXPERT, have been made generic. Developing a new application with DIAMS means essentially acquiring the application-specific knowledge, storing it in DIAMS knowledge base, parametrizing the diagnosis strategy. These tasks are rather natural since they make an intensive use of DIAMS graphical tools, for storing decision trees or subsystems models, for instance. DIAMS is really the right environment for capturing diagnosis expertise of a spacecraft project.

OPTIMUM, derived from the OPTIMUM AIV application sponsored by ESA/ESTEC and developed by MATRA MARCONI SPACE, CRI (Denmark), AIAI (UK) and PROGESPACE (Fr), is a generic product for managing complex space projects. Its main features are :

- hierarchical description of the project structure
- optimized computation of AIV plans
- advanced features to ensure the quality of the plan
- planning & scheduling rules
- assistance in monitoring and plan repair
- reporting tools
- interface with DBMS
- interactive graphical interface to edit PERT graph, hierarchical structure of activities, GANTT diagram.

OPTIMUM application to ARIANE 4 equipment bay Assembly, Integration & Validation planning /scheduling has been initialized at MATRA MARCONI SPACE.

OPSMAKER is a generic product for procedures elaboration and validation. Based on a formal computerised representation of procedures, its main features are :

- a generic kernel for operations procedures verification, formatting and execution, based on a procedure compiler
- a customized procedure editor based on a syntax-driven editor and a database
- an advanced procedure formatter for procedures printing based on a word processor
- an advanced procedure checker, based on process modelling, and providing a rich set of verifications (syntactic, local, temporal, logical,...).

OPSMAKER applications are PROCSAT (sponsored by CNES) for SPOT 4 procedures, and PREWISE (sponsored by ESA/ESTEC) for manned flight procedures (Spacelab, Columbus).

Further to the deployment of numerous applications, the emergence of generic products substantiates the OPSWARE™ domain is maturing. Hereafter, space operations implementation will not be the same !

5 - LESSONS LEARNED FROM THE DEPLOYMENT OF OPSWARE™ APPLICATIONS

As briefly described in §3, MATRA MARCONI SPACE has delivered several OPSWARE™ applications, thus giving an interesting user's feedback.

It is possible to extract some rules from this feedback :

a - OPSWARE™ systems can be implemented as extensions to existing space operations and data processing systems. Some implementations are rather straight forward (intelligent on-line documentation, failure diagnosis, procedures elaboration & validation,...). Some others may require some adaptations of the existing operations & data processing infrastructure for enabling it to integrate OPSWARE™ modules (operations execution, mission planning & scheduling,...). But the great point is that OPSWARE™ systems can be added to existing infrastructure. The evolution in the way operations tasks are performed, associated to OPSWARE™ implementation, is probably more significant (see b).

Nevertheless, some current trends in ground data systems for spacecraft control (e.g. distributed architecture, network of workstations,...) are propitious to OPSWARE™ deployment.

b - Users' motivation is crucial to OPSWARE™ implementation success. This is a key factor of success, and it comes from two facts . First, a basic requirement for OPSWARE™ systems is a perfect human-machine synergy, whatever the concerned task is. Second, if operations infrastructure is not necessarily deeply affected by OPSWARE™ deployment, it is completely different for what concerns operations tasks themselves ; they used to be performed manually, and are now performed by an "integrated human-machine intelligence" (ref 10). Users' motivation is thus mandatory !

c - A specific lifecycle is required for this type of systems. A first delivery to the user is required as soon as possible during the project, which will give a user's feedback very early, and so will guarantee the matching between system specified functionalities and the actual needs of the user. More generally, an incremental development is required to rapidly

implement changes, which may be frequent for this kind of tools. The programming technologies which are used in OPSWARE™ systems (object-oriented programming, artificial intelligence, hypertext, interface builders,...) perfectly support such a lifecycle (ref 2).

d - A more general lesson, which has been illustrated in this paper, is that OPSWARE™ is operational now, and has already deeply modified space operations philosophy, and has demonstrated a R.O.I (return on Investment).

6 - MAJOR STAKES FOR THE COMING DECADE & CONCLUSIONS

OPSWARE™ deployment is already a major fact in space operations, and OPSWARE™ stakes for the coming decade are paramount. They can be summarized in "Reengineering Space Operations" (ref. 2). What are they ?

a - Optimizing Space Missions Performances, thanks to powerful and flexible planning & scheduling systems, smarter performance analysis tools and operations support tools allowing a faster operator's reaction.

b - Reducing Space Operations Cost & Duration, thanks to operations automation, and an optimized resources management. This objective is more or less difficult to meet, depending on the nature of the space project ; the problem is completely different between a communications satellites project and a space station program, even if the objective is critical for both.

c - Enhancing Space Operations Safety, thanks to a better human-machine synergy, and to operations automation providing exhaustivity and systematization. The goal here is to reduce the number of operations errors, which still occur too much frequently in space projects, and which may have critical consequences.

d - Capturing Design Knowledge and In-Orbit Experience, thanks to the implementation of a computer-integrated technical memory and to expert tools for operations support.

These stakes are critical for the success of future space missions, and concern all the international space communities.

OPSWARE™ systems is a key to them. They can play a major role in making space operations faster, better and cheaper !

REFERENCES

- 1 - STAN DAVIS & BILL DAVIDSON
1991

2020 VISION, Transforming your business today to succeed in tomorrow's economy.

A fireside book. Published by Simon & Schister
- 2 - INFORMATION WEEK,
Bonus issue,

May 5 1992
How you fit in... Reengineering... and how you don't
- 3 - J.M. DARROY, June 1990

Knowledge-Based Systems for Spacecraft Control, Keynote Adress
1st International Symposium on Ground Data Systems for Spacecraft Control
Darmstadt, Germany
- 4 - J.FUCHS - A.GASQUET -
J. MULLER & P. VIAU,
May 1991.

The Expert Planning System Prototype : Experimentation & Users Feedback.
3rd ESA/ESTEC Workshop on Artificial Intelligence and Knowledge-Based Systems for Space
Noordwijk, Netherlands
- 5 - JY MARSON , June 1990

An Hypermedia tool for operational procedures creation and utilisation
1st International Symposium on Ground Data Systems for Spacecraft Control
Darmstadt, Germany
- 6 - JM BRENOT - Y. PARROD -
D. AUBIN & Ch PARQUET,
February 1992

ARIANEXPERT : Post-Flight exploitation of ARIANE launcher telemetries recorded during the flight.
FAIST'92 (Fielded Applications of Intelligent Software Technologies)
Toulouse, France
- 7 - JM BRENOT - Ph CALOUD &
L. VALLUY, May 1991

On the development of an operational expert system for the Telecom2 satellite control centre.
3rd ESA/ESTEC Workshop on Artificial Intelligence and Knowledge-Based Systems for Space
Noordwijk, Netherlands
- 8 - JM DARROY, March 1992

Intelligent Assistant Systems for Spacecraft Operations : the current state of the art
INFAUTOM'92
Toulouse, France
- 9 - H. LAUE, October 1990

Artificial Intelligence in Space Mission Ground Systems
41st congress of the International Astronautical Federation
Dresden, Germany
- 10 - JM DARROY , September 1990

Integrated Human-Machine Intelligence in Aerospace.
Conference introduction by the conference chairman.
3rd International Conference on "Human-Machine Interaction & Artificial Intelligence in Aeronautics & Space"
Toulouse, France

PILOT: An Intelligent Distributed Operations Support System

Arthur N. Rasmussen

The MITRE Corporation
Houston, Texas, USA

N94-23895

ABSTRACT

The Real-Time Data System (RTDS) project is exploring the application of advanced technologies to the real-time flight operations environment of the Mission Control Centers at NASA's Johnson Space Center. The system, based on a network of engineering workstations, provides services such as delivery of real time telemetry data to flight control applications. To automate the operation of this complex distributed environment, a facility called PILOT (Process Integrity Level and Operation Tracker) is being developed. PILOT comprises a set of distributed agents cooperating with a rule-based expert system; together they monitor process operation and data flows throughout the RTDS network. The goal of PILOT is to provide untended management and automated operation under user control.

Key Words: Distributed systems, artificial intelligence, automated operations

1. INTRODUCTION

The Mission Control Center at NASA's Johnson Space Center in Houston has been the focus of manned space flight operations for nearly 30 years. The Center's current responsibility is ground support for all Space Shuttle missions; in the near future this will be expanded to other manned space activities including Space Station *Freedom*. The overall goal of the flight controllers in Mission Control is to ensure a safe flight that achieves the highest level of mission objectives.

An important subgoal is performing the ground support role in an economically efficient manner. Several approaches to this are being pursued, one of which is the exploitation of new technologies to increase the level of automation. The Real Time Data System (RTDS) project is chartered with exploring the applicability of new technologies, proving concepts, and demonstrating their practical application to solving problems of real-time operations in Mission Control (Refs. 1-3). An important aspect of this work is the evaluation and integration of distributed system technologies into

the unique Mission Control environment.

An area that exemplifies the shift to distributed solutions is the RTDS data acquisition and distribution system. This system ingests the real time telemetry stream received from the Shuttle, decommutates it, applies transformations such as calibration to engineering units, and transfers the data to a set of engineering workstations where the data is analyzed by flight control application software. The complex telemetry stream is composed of a main stream and several embedded asynchronous substreams containing thousands of data being collected at a variety of rates and having hundreds of configuration states.

The RTDS solution incorporates a commercially available programmable telemetry processor which is augmented by custom integration software operating in a workstation. Figure 1 shows an overview of this approach. The approach is representative of architectures being adopted for use in Mission Control and has numerous advantages over previous approaches.

A goal of RTDS is to raise reliability to the highest affordable level. Key to reliability is uninterrupted service from the application program's viewpoint. This allows for failure of components that are providing a service if a compensating action minimizes the impact on the application. Since RTDS is based on commercial hardware and software platforms (including the operating system and base networking software), services such as telemetry data delivery must capitalize on the platforms' strengths and compensate for weaknesses.

The application services provided within RTDS are supplied by custom software components that bridge the gap between the needs of the applications software and the services available as part of the basic platform. Until recently, reliability was obtained by employing a human operator to monitor the health of the system services, diagnose problems and take manual corrective action to correct them. However, the slow operator response to multiple failures and the cost of continuous staffing were not acceptable in the long term.

2. APPROACH

To provide better, faster service and to release the human operators from monitoring dozens of workstations and their software processes, a decision was made to automate with the goal of eliminating the need for a human operator. At the same time it was recognized that the human operators held considerable expertise on efficient operation and that capturing this expertise would be very desirable.

2.1 Design Philosophy

The design philosophy adopted for the task attempts to exploit the strengths of the distributed environment rather than to force the imposition of previous (primarily centralized) architectures. One implication is that problems should be dealt with locally whenever possible so as to provide the quickest response and the least impact on other elements. A second design precept assumes that failures will occur, and that expedient recovery is pragmatically more feasible and desirable than taking extraordinary measures to ensure that a failure will not occur. The third principle states that, wherever possible, the custom software elements should make the needed accommodations since it would be neither sensible nor economically feasible to alter the commercial components of the system.

Within this design framework rests an approach based on three main elements. The first element is in the implementation of the custom software components that provide the application services. These are designed so that they are easily replaced at execution time by a second instance should the first instance fail. In addition, they are instrumented to make their operation externally visible. This allows another process to monitor their states, control their behavior, gather information when failures occur, and diagnose problems.

The second element in the approach is a process that monitors each service delivery process, controlling its behavior, tracking its operation and replacing it should it misbehave. An instance of this process runs in the local environment of each workstation and has the responsibility of maintaining the integrity of services within the environment. The process is called the autopilot because its operation is like that of the autopilot of an aircraft; the software "knobs" on the autopilot are set and the autopilot does its best within its range of control to maintain the requested state.

Whenever there are a set of related entities operating in a distributed environment, issues usually arise

that are beyond the scope of a single entity. Means for dealing with such issues are a current area of research; the initial approach for RTDS is the proven technique of having a central coordination entity. As a result the third element in the design is a rule-based expert system that oversees the operation of the autopilots; this is named PILOT (Process Integrity Level and Operation Tracker).

To act as its representative in each workstation's local environment, PILOT employs a process called the "spy". The spy gathers information from the local environment, filters it according to the wishes of PILOT, and transfers it to the PILOT over the network. It also provides PILOT with the mechanism to advise and control local processes.

2.2 The Service Delivery Processes

Nearly all of the processes that provide RTDS services to applications operate as "daemons", executing in a background mode unattached to any specific user interface. In addition, the human user is generally uninterested in (and often unaware of) the details of daemon operation. To provide visibility into daemon behavior and to provide a means to control daemon behavior, a standardized mechanism has been developed and integrated into all RTDS daemons.

This communication mechanism, nicknamed "VIZ", makes it possible to inspect a running process to determine detailed information about its state, configuration and recent error history. This allows a monitoring process such as autopilot to closely watch each daemon and determine whether the daemon is performing useful work or has failed in some way (e.g., has deadlocked or otherwise gotten "stuck"). The monitoring process cannot only quickly detect a failure, but also take actions such as logging the daemon's state information for later analysis or even predicting failure before it actually occurs. Other uses for the visibility features include providing a simple display of daemon operation to users and a more detailed one for development purposes.

The action request portion of VIZ allows an external process to make requests of the daemon. The initial use of this feature provides a simple means to request the process to terminate; using the VIZ capability overcomes the normal operating system restriction that inhibits one user from terminating the processes started by another user. A more important use is the ability for a monitoring process to request status information from a daemon. A planned use by the autopilot process is to provide hints to a new instantiation about the

circumstances of the demise of its predecessor; this may allow the replacement process to avoid the problems encountered by its predecessor.

The VIZ capabilities appear to the daemon to be a set of procedures that allow the daemon to register itself under its name, update its state information and receive action requests. The current implementation is based on transformation of most procedures into direct accesses to a shared memory area. This reduces the overhead to a negligible amount while preserving the ability to make enhancements. The information is stored in a canonical form that permits transporting it over the network in a transparent manner.

2.3 The Autopilot Process

The capabilities of the autopilot form the first level of system management and reflect the basic design tenet that situations arising on a workstation should be handled locally if possible. The purpose of autopilot is to manage the local process environment, track process operation, record anomalous behavior, and replace misbehaving processes with new instances. The normal operation is as a daemon in a "set and forget" mode; one autopilot executes on each workstation, receiving instructions as to the desired process environment and taking action to create and maintain that environment.

In normal operation the autopilot manages all of the daemon processes executing within the workstation. It starts the processes and monitors their health and execution status using both the VIZ and operating system capabilities. Figure 2 shows how the autopilot fits into the workstation's environment. If it detects a problem in a daemon, it issues a warning to the daemon using an appropriate mechanism. After a suitable interval, autopilot removes the process and replaces it with a new instance. If the daemon is capable, the autopilot issues a "soft" terminate request to allow the process to perform cleanup before terminating. If the process fails to terminate, autopilot forces termination. The state information of failed processes and all actions taken on processes are logged by autopilot for later analysis.

The actions taken by autopilot can be tuned in several ways. Each process managed by an autopilot has an associated set of management policies and parameters that determine how the autopilot monitors and takes actions on the process. Examples include the means and criteria by which the autopilot judges the health of the process, the steps the autopilot performs while terminating the

process, and relationships among related managed processes. A managed process can interact with the autopilot in controlled ways, including advisory information and requests to the autopilot to adjust its management policies to reflect activities being performed within the process. For example, a managed process can advise the autopilot that it is entering a phase in which it will appear dormant for an unusually long period.

Like all RTDS daemons, an autopilot reports on its state using the VIZ capabilities. This permits several autopilots to operate and manage independent sets of processes while being managed by a parent autopilot. The state information from VIZ is also used by an interface process (the autopilot "control panel") whose purpose is to control the autopilot. The process has the abilities to determine which processes are being managed by the autopilot, which policies are in effect, and the policy-related state of each process. The interface process also provides a mechanism for making requests such as starting the management of new processes, restarting or removing currently-managed processes, or requesting the termination of the autopilot.

2.4 The PILOT Expert System

Early experience with using the autopilot to manage workstations clearly illuminated which areas were well-addressed and which were not. Once set into operation with an appropriate configuration, an autopilot manages local operations well. As expected, the autopilot easily handles stuck processes and unexpected process terminations. Its operational shortcomings are directly traceable to its lack of contextual and global knowledge; it operates having little information about the expectations of the user and the general status of the network and other workstations.

The role originally envisioned for PILOT, to replace the human system monitor, has been validated by experience with the autopilot. PILOT is still in active development under the strategy of capturing the expertise and reducing the need for human intervention one functional area at a time. Two primary goals are being used to decide the order in which PILOT capabilities are developed. The first is which capabilities will release the maximal amount of the human resource, and the second is which are required as foundations for other capabilities.

The current PILOT is primarily an advisor. It builds and maintains a dynamic model of the workstations, their internal states and their

interrelationships. It creates several basic displays, including a diagram of the current data flow hierarchy as shown in figure 3, and displays of the detailed status of an individual workstation. It reduces the workstation state information to a small set of qualitative parameters. These are now used to characterize system state for the human operator; they are also being used to construct a model-based reasoning capability that will perform diagnostic tasks such as isolating workstations or network segments that are causing loss of data to a subtree.

PILOT is implemented as a rule-based expert system using Gensym Corporation's G2™ real-time expert system shell. The shell provides a user extensible object and class hierarchy, a graphical user interface, procedures and rules. Due to licensing restrictions, PILOT operates on a dedicated workstation. The general strategy implemented in the rules is to detect events and activate rules and procedures to evaluate them.

2.5 The Spy Process

To obtain information and cause actions on a workstation, PILOT uses a spy process as its agent in the local environment of each workstation that PILOT is monitoring. The spy process executes in the local environment and communicates over the network with PILOT. A spy appears to the local environment as a daemon and is managed by the autopilot running in that environment. It attaches to system services, the VIZ service and the program interface provided by autopilot in order to monitor activity in the local environment. Figure 4 shows the role of the spy process.

Each spy gathers information on the local environment and sends it to PILOT; from PILOT it receives requests directing the spy to adjust its behavior or to interact with other processes such as autopilot. PILOT can request that the spy send automatic periodic updates or can place it in a polled send-on-demand mode. A spy attempts to minimize its network traffic by performing local filtering and by sending only changed value events. Current work on the spy agent emphasizes increasing its intelligence; e.g., a spy will know how to initiate and track multi-step operations that are activated by a single request from PILOT.

3. FUTURE DIRECTIONS

As PILOT masters the simpler system management tasks, it will be expanded in two areas. The first is overall system setup and configuration. Currently the human operator provides information about the day's activities and the configurations needed to

support them. This process will be automated so that PILOT is aware of the day's schedule and can advise the workstations on proper configuration. A user advocate process is currently under development that will allow the local workstation user to make simple requests to perform expected tasks (e.g., "set up my workstation to support the scheduled simulated mission") and to provide for special usage (e.g., "I want to review the data from a specific previous mission"). This process will establish the desired configuration and inform PILOT of the user's desires so that, for example, PILOT does not assume the workstation is improperly configured.

The second major area of work will be in dynamic monitoring and control of the system as a whole. Issues at a global level will be addressed such as the policies and procedures used to reroute data flow around failed workstations or network segments. Capabilities will be added to PILOT to perform active troubleshooting and corrective actions. Another area of investigation will be the use of contextual information adjust policies and the ability to learn appropriate policies based on experience.

The planned strategy is to experiment with techniques in PILOT to gain experience quickly, and then to distribute the knowledge gained to the lowest level possible (e.g., to the spy or autopilot). Although the current architecture is strictly hierarchical with PILOT at the apex, plans are to introduce peer-to-peer communication; the autopilots will be first augmented in this way. PILOT will be retained to serve as a central decision resource as well as provide a system-level troubleshooting interface for a human operator. As the network expands and as motivated by the nature of flight control, clusters of workstations will be organized to provide services to flight control areas. This may require several PILOTs operating cooperatively to maintain their clusters and negotiate with neighbors for resources and information.

4. REFERENCES

1. Muratore, John F. et al. 1990. "Real-time Data Acquisition in Mission Control." *Communications of ACM*, 33, 12: 18-31.
2. Heindel, Troy A. et al., 1991. "Knowledge-based Systems in Space Shuttle Mission Control." *Aerospace America*, 1991, 8-11.
3. Muratore, John F. et al., 1989. "Real-time Data Acquisition for Expert Systems in Unix Workstations at Space Shuttle Mission Control." *Twenty-first Symposium Proceedings, Society of Flight Test Engineers*.

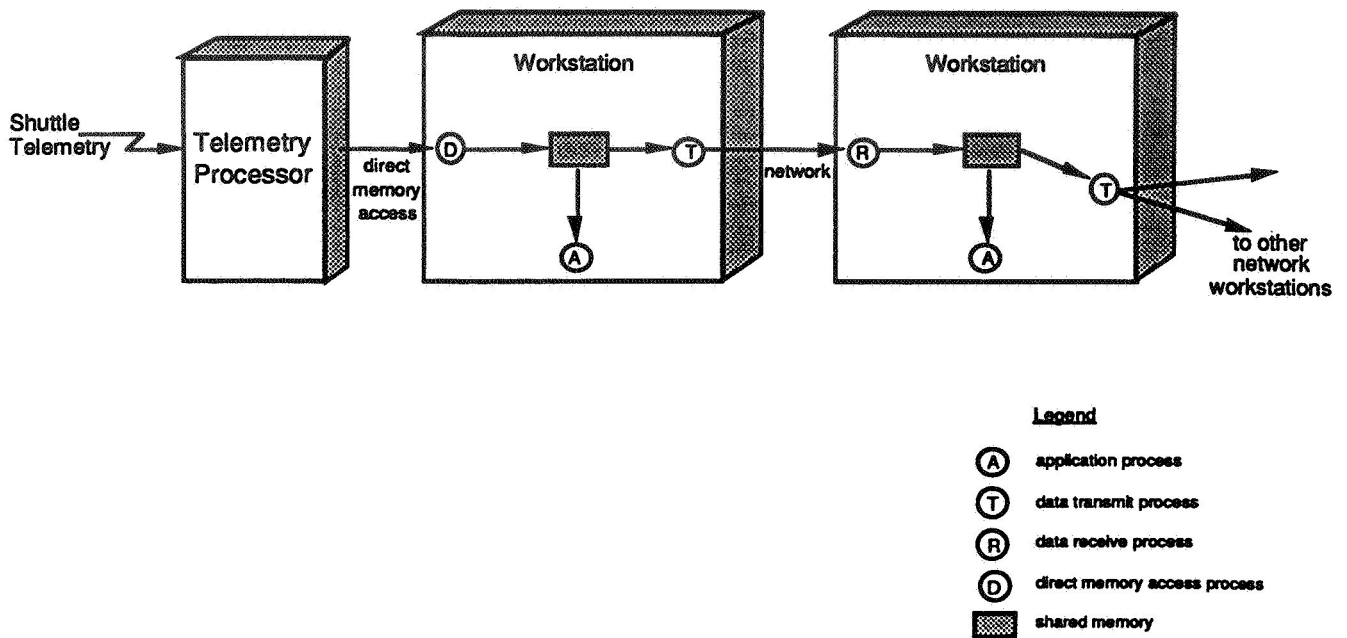


Figure 1. RTDS Telemetry Data Flow

Workstation Environment

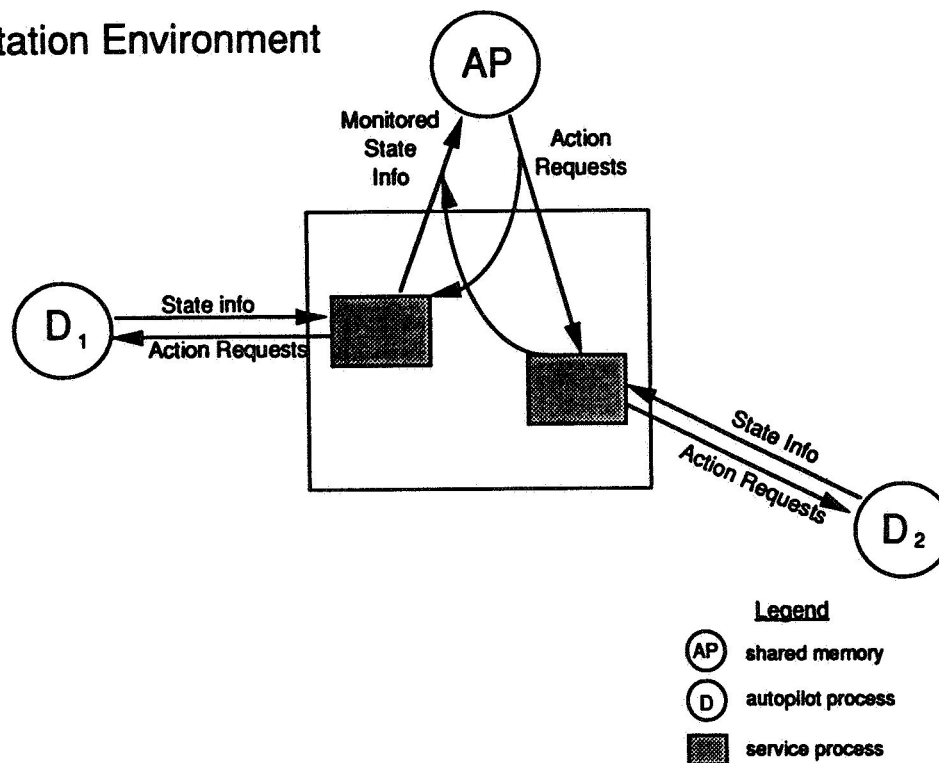


Figure 2. Autopilot Operation

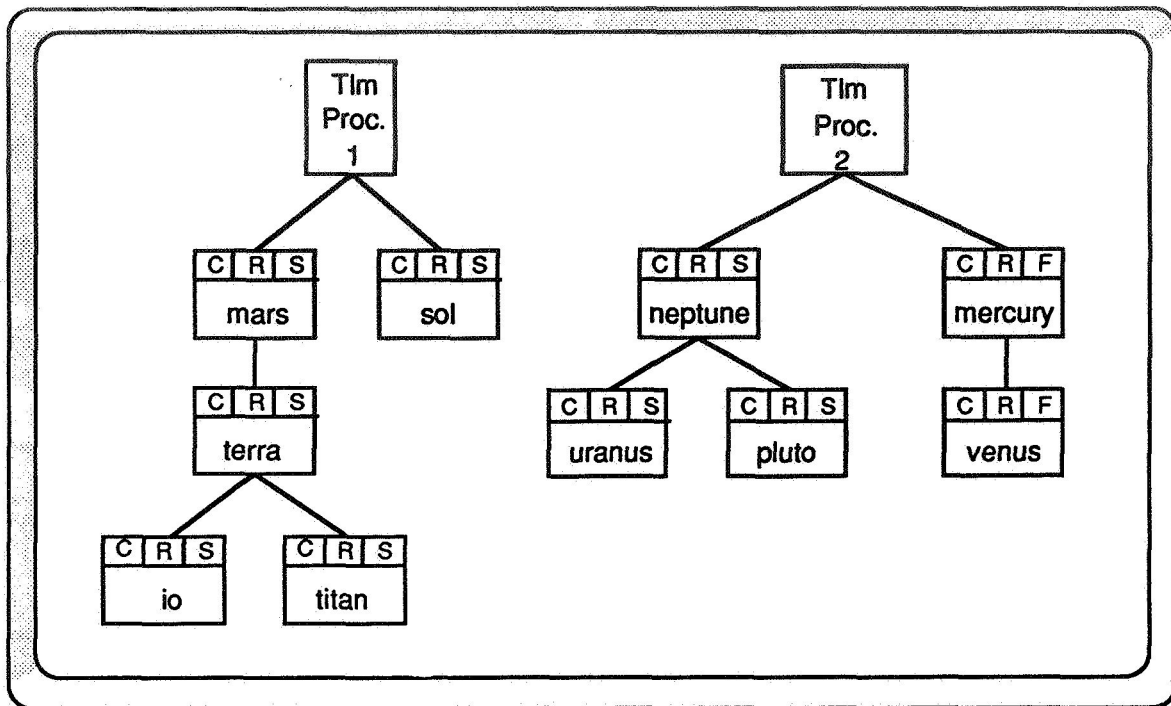


Figure 3. PILOT's Network Hierarchy Display

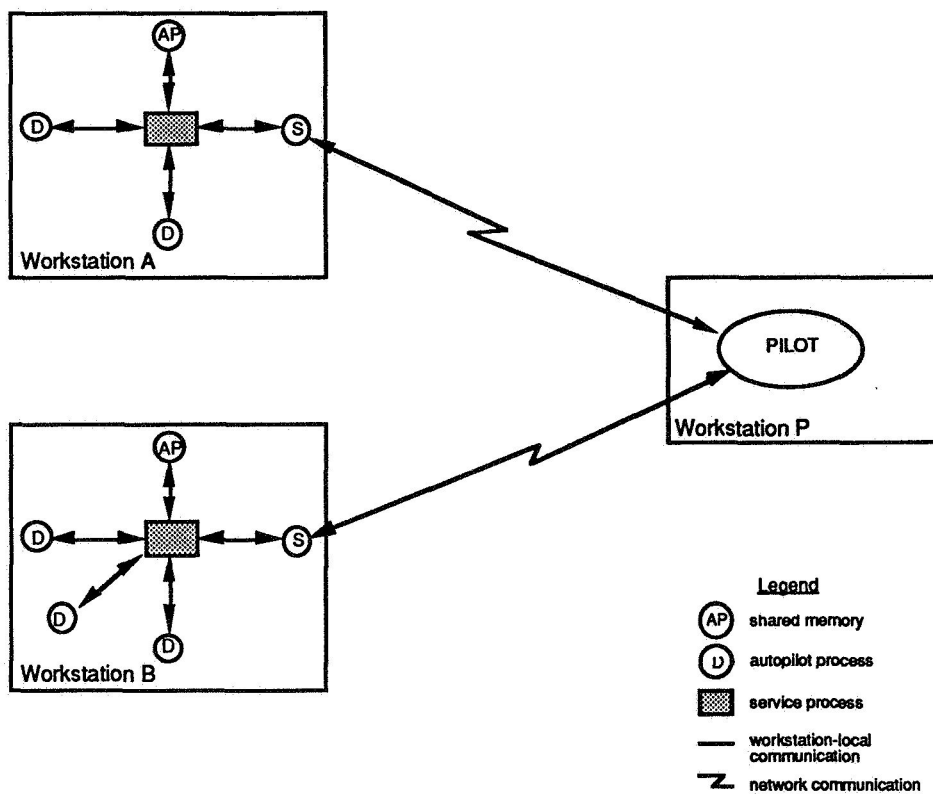


Figure 4. Spy Process Operation

ESSOPE: TOWARDS S/C OPERATIONS WITH REACTIVE SCHEDULE PLANNING

J Wheadon

N 9 4 - 2 3 8 9 6

ESA/ESOC European Space Operations Centre
Darmstadt, Germany

ABSTRACT

The ESSOPE is a prototype front-end tool running on a Sun workstation and interfacing to ESOC's MSSS spacecraft control system for the exchange of telecommand requests (to MSSS) and telemetry reports (from MSSS). ESSOPE combines an operations Planner-Scheduler, with a Schedule Execution Control function. Using an internal "model" of the spacecraft, the Planner generates a schedule based on utilisation requests for a variety of payload services by a community of Olympus users, and incorporating certain housekeeping operations. Conflicts based on operational constraints are automatically resolved, by employing one of several available strategies. The schedule is passed to the execution function which drives MSSS to perform it. When the schedule can no longer be met, either because the operator interferes (by delays or changes of requirements), or because ESSOPE has recognised some spacecraft anomalies, the Planner produces a modified schedule maintaining the on-going procedures as far as consistent with the new constraints or requirements.

1. INTRODUCTION

Many modern spacecraft, particularly those for science missions or experimental technology missions, are characterised by carrying payloads comprising a number of subsystems which will be employed on a variety of tasks or experiments. Recent examples of such ESA spacecraft are ERS-1, EURECA, and in the field of telecommunications, OLYMPUS.

The operation of such a spacecraft is a complex undertaking; there are usually many constraints which have to be respected. Such constraints may prohibit the simultaneous or concurrent operation of two or more given payload subsystems in certain specific modes, for example because of interference or disturbances caused by one to another, or because of the shared use of some resource, e.g. a

transponder, or a supply of electric power, or some available bandwidth in a downlink signal.

Whereas it has been the traditional practice to operate spacecraft following rigidly pre-defined and pre-tested Flight Control (procedural) Timelines, and indeed this practice is generally continued for spacecraft platform operations, for complex missions such as those mentioned above a more flexible approach to defining the activities of payload operations is needed. Not only must the "on-board" constraints be respected, but also the temporal requirements of the satellite users, e.g. Principal Investigators or Experimenter Institutes, have to be accommodated. For example, an Earth observation satellite's payload will be constrained to operate in one or another mode at different times according to the nature of the target areas (sea, land, ice...) which the satellite is passing over. As another example, users of a telecoms satellite may wish to schedule use of a given payload at times when they can have the use of specific ground station equipment for the performance of a communications experiment.

Because of this need for flexibility in defining the activities of day-to-day operation, there is much interest in having the use of automated planning systems to generate operational timelines or schedules which define the actual activities in detail. Such systems can generate the sequences of telecommands which have to be uplinked to the payload at given times, or which alternatively will be loaded into an on-board Master Schedule for automatic on-board execution later when required.

Automated "mission planning systems" have already been developed and put into daily operation for example for EURECA and ERS-1, where the schedules are prepared several hours in advance of their execution.

It was decided to investigate the possibility of linking more closely the planning/scheduling function with a ground-based automatic schedule

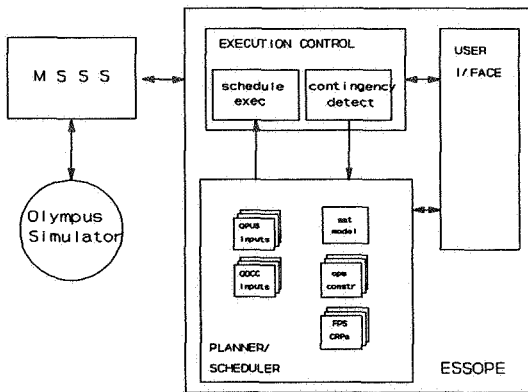


Figure 1: System Overview

execution function. It was also decided to attempt to introduce an element of "reactive planning", i.e. fast revision of an existing currently executed schedule in the event of new constraints or user demands arising more or less in real time, or of on-board or other anomalies being detected during the operation and preventing some of the scheduled activities.

In order to be able to demonstrate the concept of closely-coupled reactive planning and execution of operations, the Olympus geosynchronous telecommunications satellite was chosen as a "test case", and a demonstration system architecture was conceived which made use of existing ESOC infrastructure, namely the Multi-Mission Support System (MSSS) and ESOC's real-time operations simulator of Olympus. The MSSS was augmented by the ESSOPE, which provides an "intelligent front-end" for the user, performing detailed schedule planning of the operations activities, and online step-by-step control of their execution.

Figure 1 shows the combined configuration, and its operation is described later in this paper. But first follows a summary of the Olympus planning and execution problem domain, in order to give the reader an appreciation of the technical motivation for the choice of Olympus as the "test case" for the ESSOPE study.

2. THE PROBLEM DOMAIN

Note: The description given below relates to Olympus as it was being operated at the time when ESSOPE was being developed (1989-1991). In early 1991 Olympus suffered a solar array failure which has resulted in a significant reduction in the available power supplied to the payload.

Olympus is a large geostationary telecommunications satellite which was launched in July 1989. It carries a multipurpose experimental payload to test future telecommunications flight hardware and provides an opportunity for communications and broadcasting experiments throughout Europe, and also in Canada.

There are four quite separate payloads on Olympus-1, each with its own antennas:

- the Direct Broadcast Service (TVP) Payload
- the Specialised Services Payload (SSP)
- the 20/30 GHz Advanced Communications Payload (CMP)
- the Propagation Payload.

The exploitation of these payloads in orbit is co-ordinated under an overall Olympus Utilisation Programme, which encompasses all aspects of the satellite's use.

The TVP payload operates at 18 GHz for uplink and 12 GHz downlink and there are two channels each with a pair of redundant high-power TWT amplifiers feeding two separately-steerable antennas which can beam down over different regions of Europe, Scandinavia and North Africa. A wide-beam uplink antenna can be accessed from anywhere in Europe. The TVP payload supports experiments in direct sound and TV broadcasting for domestic reception, education and training, and interactive information services, as well as technical tests of various kinds (TV standards, HDTV, antenna measurement techniques), for more than 20 user organisations.

The Specialised Services Payload (SSP) offers bandwidths of 18 or 36 MHz, and operates in frequency bands between 12.5 - 19.3 GHz. Its antennas provide an array of five circular overlapping beams in a fixed pattern steerable over a very wide area. The SSP supports four up- and downlink signals which can be switched between the five beams, either in static mode, or dynamically allowing satellite-switched TDMA experiments to be made.

Experiments cover a range of applications and technical tests, including SS-TDMA, tele-education, high-quality document image distribution, compressed video, news-gathering and frequency diversibility for many user organisations.

The Advanced Communications Payload (CMP) provides two 40 MHz bandwidth channels operating in the 30 GHz (uplink) and 20 GHz (downlink) regions. An alternative wideband capability of 700 MHz is supported. There are two spot beams independently steerable over a very wide area. Three output amplifiers provide one-for-two redundancy.

A large range of technology and applications experiments are supported by the CMP for more than 30 user organisations. Worthy of special note is the Inter-Orbit Communications experiment in space data relay being successfully performed by ESA. This involves continuously steering on of the Olympus 20/30GHz payload antennas to track and communicate with EURECA, ESA's low-orbit Retrievable Carrier satellite, which has an orbit inclination of about 27° and a period of about 90 minutes.

The Propagation Payload just generates three signal beacons. Its operation is very simple, and was not sufficiently interesting to be considered in the ESSOPE study.

On any typical day, many user institutes (here called "experimenters") are able to make use of Olympus services, however, not all can be accommodated simultaneously. ESA therefore performs scheduling of all operations, nominally about a week in advance. However, it quite often happens that adjustments to the schedule have to be made at quite short notice: a few hours is not uncommon. The scheduling is done using simple spreadsheet and database tools, with resource conflicts being resolved manually. In fact this system is quite adequate for normal weekly scheduling, but at the same time the Olympus scheduling problem is sufficiently complex to make it interesting as the "test case" for ESSOPE, especially for demonstration of "reactive planning" which cannot be achieved to anywhere near the same degree of real-time reaction by manual methods.

Scheduling takes into account requests for specific Experiments (characterised by Experimenter, payload and its configuration) to be supported during given time windows. It must also take account of certain platform operations which can affect the availability or usability of certain payload configurations.

Platform operations, which are essential to the health of the spacecraft, take priority over experiments which may conflict with them. Such platform operations include: Eclipse operation (reduced power available), station-keeping, attitude manoeuvres, and certain AOCS (Altitude/Orbit Control System) gyro calibration checks; the resource conflict created by the latter is not on-board, but on-ground: It keeps the spacecraft controller fully occupied for half an hour, so that he is not able to spend time configuring payloads ready for the next customer.

Concerning payload resource constraints: Apart from the need to direct the required signal beams to the right places, and the need to allocate channels to users, some mutual resource constraints exist between payloads; for example, both the SSP and CMP share the 20/30 GHz transponders for some experiments. There are also power availability constraints to be respected, which become more severe in eclipse.

Concerning payload configuration, the time to arrive at a desired new configuration will depend on what configuration was used immediately beforehand, and this must be taken into account by the scheduler. The number of intermediate states, through which the payload must be switched, directly affects the time to go from state A to a different state B. Usually each intermediate step is effected by a single telecommand (or block-command), and these must each be verified (verification delay 20-30 seconds per command) before proceeding to the next intermediate state. In certain cases longer delays and pauses have to be planned in the intermediate states. For example, the TWT amplifiers are normally kept warm either by active heating, or by the processing of signal traffic, but it is sometimes necessary to switch them off completely to conserve power. Subsequent pre-warmup to bring them back into use can take half an hour or so.

However, because most of the possible payload reconfigurations can proceed at a rate limited only by the telecommand verification process on the ground (send command, await verification by MSSS on next arriving TM format... nominal delay 18 seconds), and because such step-by-step operations (on the real Olympus, without ESSOPE) effectively tie up the spacecraft controller full-time, it the practice in those cases to avoid concurrent reconfiguration of more than one payload at a time.

This is one more "operating constraint" applied by ESSOPE, which is otherwise capable of scheduling and also executing reconfigurations of the payloads in parallel. In cases involving intermediate step delays, multiple payload reconfigurations actually do proceed in parallel.

In addition to operating constraints imposed by the satellite itself, some ground-based factors have to be considered during scheduling, mainly related to the need to coordinate the activities of the experimenters in real time. This last point basically requires the ESA Olympus operations coordination function to contact experimenters by phone to ensure that they will be ready to use their allocated time slots, and that they do not transmit to Olympus outside those slots. This coordinator is also a schedulable resource; basically the available staffing allows phone calls to only one experimenter at a time.

3. OPERATIONS WITH ESSOPE

The ESSOPE has been developed to run on a SUN workstation, and is linked via a specially designed interface protocol to the MSSS software. The spacecraft operator is served through an ESSOPE user interface on the SUN, in addition to his normal MSSS operations interface screens/keyboards.

ESSOPE provides two main operating functions to the user: A Planner/Scheduler which generates a schedule of Olympus operations, and a schedule implementation function ("Execution Control") which controls the execution of the schedule, coordinating it with the state of the spacecraft telemetry and with the activities of the spacecraft operator. The Execution Control function is served by the MSSS to perform the basic control functions on the spacecraft (simulator !).

ESSOPE also provides appropriate knowledge acquisition functions, to configure it off-line (eg to allow the definition of procedure steps). These knowledge acquisition functions are not further described in this paper. For an exhaustive account of the design and functioning of ESSOPE, see (Refs. 1,2).

ESSOPE Execution Control effectively takes over some of the low-level mechanical tasks of the spacecraft operator. Thus he no longer has to transcribe manually into MSSS the telecommands

described in paper-based procedures or timelines; instead, ESSOPE reads these directly off the schedule generated by its planner/scheduler, and sends the appropriate telecommand requests to MSSS via the interface. In return, MSSS provides ESSOPE with reports confirming the uplink of the telecommands, and subsequently also their verification based on simple direct responses observed in the spacecraft telemetry.

In normal MSSS operation (without ESSOPE), the operator is required to perform visual checks of spacecraft status at each step of the procedure, by viewing displayed groups of telemetry parameters on the MSSS screens. He has to compare the displayed values with an equivalent hardcopy sheet of required values, included in the standard procedures handbook: The "Flight Operations Plan". ESSOPE is able to make these additional checks automatically for him, at the same time advising him which MSSS displays to call up to monitor the same information. The required telemetry parameters for these checks are specified to MSSS by ESSOPE directly over the interface, and MSSS responds by sending regular messages containing the latest values, back to ESSOPE.

MSSS performs continual limit checks on a large number of telemetry parameters, and when out-of-limits are detected, raises out-of-limit alarms on the MSSS user displays. When ESSOPE is in use, the out-of-limit information is sent to it by MSSS, and ESSOPE is able to perform some simple "first-level" evaluation of the anomaly, and propose to the operator an appropriate Contingency Recovery Procedure (CRP).

The principal regions of the ESSOPE operator interface for Ops Execution Monitoring are shown

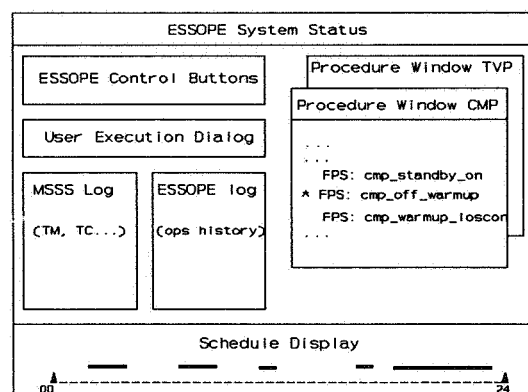
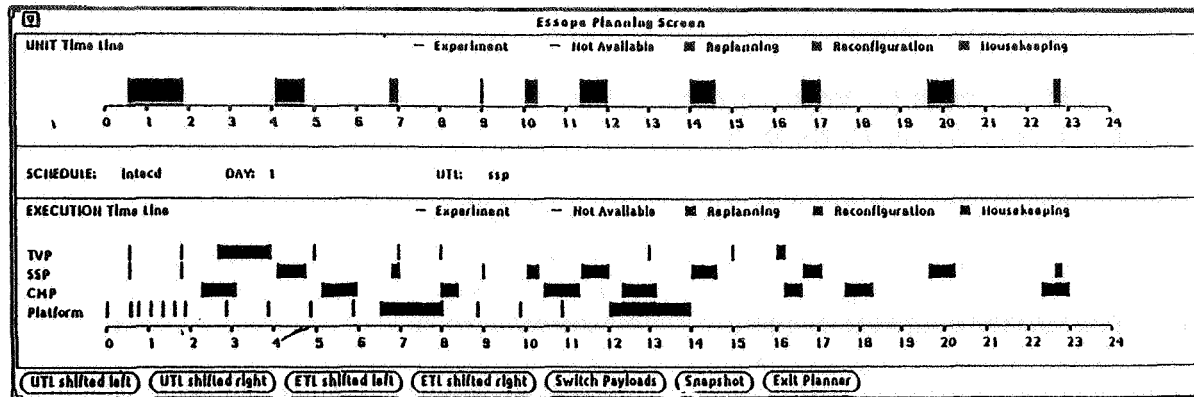


Figure 2: Execution User I/F



1Figure 3: Planner Screen with Schedule Display

schematically in Fig. 2. An overview of the schedule and of the steps of each active "procedure" in execution are displayed, with markers at the current position in each case, so the operator can follow the progress exactly. Note that a payload "procedure" in ESSOPE terms is not exactly equivalent to a Flight Control Procedure (FCP) in the traditional sense meaning a predefined, documented and validated procedure which the operator reads step-by-step out of a book on his console. A "procedure" for ESSOPE means a sequence of actions needed to change one of the payloads from one operating mode to another, *in the context of the current schedule*. This point is elaborated later under "Planning/Scheduling".

The schedule display shows a planned timeline for each payload and for the platform. Each payload timeline defines a series of "procedures" for switching the payload configuration, chained together with the periods of steady-state payload operation where the actual experiments are performed. ESSOPE is not able to build all possible platform timelines, but a representative set of procedures for the platform can be generated.

At any time, more than one procedure may be active concurrently, according to the progress along the individual timelines. of the schedule. The operator is no longer required to enter the required telecommands (or call up the TC sequences) manually on MSSS; instead, the appropriate commands for each procedure step are sent automatically into MSSS by ESSOPE. The operator can thus retain a higher-level view of the operation with less distraction, and still has at his disposal all standard MSSS information displays.

Due to the modular architecture of the MSSS software, virtually no changes to it were necessary to accommodate ESSOPE, apart from the addition of a special interface server module. In fact the MSSS is essentially unaware that part of the operator's normal (low-level) functions have been taken over by ESSOPE.

4. PLANNING/SCHEDULING

Fig 3 shows the schedule as displayed to the user. There are basically four parallel timelines, three for the payloads and one for the platform.

The Planner/Scheduler generates schedule segments of 24 hours at a time, and can cover up to seven successive days. The starting state for each new day is automatically inherited from the final state at the end of the previous day. The operator specifies the requirements in the form of Platform Operations and times (these are considered immovable), and Experimenter requests. The latter specify nominal start and finish times, and acceptable variations on these, as well as some parametric data affecting settings for the required payload configuration.

Earlier in this paper, the concept of a "procedure" was briefly introduced. By "procedure" is meant a sequence of actions, or in the terminology of AI, "a goal-oriented plan" of actions which have to be performed *in the context of the current schedule* on one given hardware unit or subsystem, to change it from a starting state to the desired goal state.

The planning of each timeline is derived from knowledge of an internal state-based model of the

relevant spacecraft subsystem. Each procedure in the timeline is composed of "Flight Procedure Steps" (FPS), which are placed at the appropriate points on the timeline. These are the smallest schedulable unit of activity, are rigidly-defined "mini-procedures", and are classified according to their function; a "state-to-state" FPS will move the state of a payload subsystem from one node to an adjacent node in the model's state-space by sending telecommands. Other classes of FPS exist, eg a "wait" FPS can request and await some information from the operator, or can delay until some telemetry parameter has acquired a given value. Each FPS is given a unique name to indicate its function, and the sequence of FPS names for each active procedure is displayed by the Execution user interface in a "procedure subwindow".

The ESSOPE Planner/Scheduler builds the schedule in such a way that the timelines are only loosely-coordinated. And may be executed concurrently with no greater mutual synchronisation than is necessary. The scheduler respects all the operational constraints, and all conflicts discovered are automatically resolved by applying a number of strategies, eg shortening experiments or shifting them within specified time limits. Some of the factors which the conflict resolution takes account of are experiment priority, Experimenter priority, the proximity in time of experiments, the actual duration of the procedure required to switch each payload between the required modes. The user has the option to influence, during the scheduling process, the choice of strategy to be applied whenever there is a conflict, or he can leave the scheduling process to run automatically. Any redundant FPS are deleted. Production of a 24-hour schedule segment is quite fast, being done in about 2 to 3 minutes on a Sun SPARC-2.

The schedule is passed to the Execution controller as a set of parallel streams of FPS grouped into procedures, one stream per timeline. At the moment of execution, the operator has the choice of running any individual procedure in automatic mode, or in user-authorise mode where ESSOPE awaits his authorisation to commence each new procedure step. The schedule specifies an earliest and latest time for each FPS, and any FPS may be flagged with pre-conditions to block its execution, and post-conditions which it shall establish to affect the pre-conditions of other FPS (either subsequent FPS of its own thread or FPS of others); in this

way a dynamic and elastic coordination between the timelines is achieved.

5. REPLANNING

It is possible that accumulated delays in execution of a given thread result in an inability to meet the final deadline for the next FPS. Or it may be necessary to abandon the execution of a particular thread or prematurely terminate the execution of an experiment because of the need to execute a Contingency Recovery Procedure to deal with a spacecraft anomaly. Or an Experimenter may announce at short notice that his requirements for his service that day have changed (maybe his equipment has broken down, for example). These situations lead to the need for a re-planning operation which is performed by the Planner/Scheduler; this is done while allowing the continuation of all active timelines which do not have to be stopped due to the problem in hand. The Planner/Scheduler passes the new version of the schedule (new timelines) to the Execution Controller which adopts it for continued operations.

6. REFERENCES

1. Gusmeroli S, Pezzinga A, Barone S, Briatico D, Wheadon J. 1990. ESSOPE - Expert System for Spacecraft Operations planning and Execution. In *Proc Symp on Ground Data Systems for S/C Control*. Darmstadt, Germany. ESA SP-308.
2. Gusmeroli S, Monti M. 1991. A Knowledge Based Assistant for Olympus Spacecraft Operations. In *Proc Artificial Intelligence and Knowledge-Based Systems for Space Workshop*. ESTEC, Noordwijk, NL.

7. ACKNOWLEDGEMENTS

ESSOPE was created as part of an ESA-funded study. Thanks are due to: A Pezzinga, S Gusmeroli, and M Monti of FIAR, Milan (prime contractor) for problem domain analysis, architecture design, and the development of the Planner/Scheduler; also to D Briatico, and S Barone of INTECS, Rome, for the development of the Execution Controller and MSSS Interface. The knowledge acquisition editors were developed by both companies.

VERIFICATION AND VALIDATION OF EXPERT SYSTEMS USED IN GROUND DATA
SYSTEMS

C. Culbert
NASA/JSC
Houston, Texas, USA

*Paper presented but not submitted to proceedings
as of December 14, 1992.*

The ADVANCED TECHNOLOGY OPERATIONS SYSTEM ATOS

J.-F. Kaufeler¹, H.A. Laue¹, K. Poulter², H. Smith³

N94-23897

Abstract

Mission control systems supporting new space missions face ever-increasing requirements in terms of functionality, performance, reliability and efficiency. Modern data processing technology is providing the means to meet these requirements in new systems under development.

During the past few years the European Space Operations Centre (ESOC) of the European Space Agency (ESA) has carried out a number of projects to demonstrate the feasibility of using advanced software technology, in particular, knowledge based systems, to support mission operations.

A number of advances which must be achieved before these techniques can be moved towards operational use in future missions, namely, integration of the applications into a single system framework and generalisation of the applications so that they are mission independent.

In order to achieve this goal, ESA has initiated the Advanced Technology Operations System (ATOS) programme, which will develop the infrastructure to support advanced software technology in mission operations, and provide applications modules to initially support: Mission Preparation, Mission Planning, Computer Assisted Operations and Advanced Training.

The first phase of the ATOS programme is tasked with the goal of designing and prototyping the necessary system infrastructure to support the rest of the programme. This paper presents the major components of the ATOS architecture.

This architecture relies on the concept of a Mission Information Base (MIB) as the repository for all information and knowledge which will be used by the advanced application modules in future mission control systems. The MIB is being designed to exploit the latest in database and knowledge representation technology in an open and distributed system.

In conclusion we present the technological and implementation challenges we expect to encounter, as well as the future plans and time scale of the project.

between a spacecraft in orbit and the human user of the spacecraft on ground. It is essential to conduct a space mission and to fulfil the mission objective. These objectives are met by:

- mission planning
- monitoring and control of the spacecraft platform
- planning and execution of flight dynamics manoeuvres
- configuration of platform and payloads
- monitoring and control of payloads
- generation of operation schedules
- derivation of spacecraft engineering and navigational state
- reception, processing, annotation and distribution of payload data (in case of measuring payloads)

The scope of a mission control system can range from a single expert working with a small desktop computer to a string of control centres with multiple spacecraft control processing systems, communication systems and stacks of documentation. Leaving aside the system required to communicate with the spacecraft on one hand, and with other ground facilities or mission users on the other hand, a mission control system has these essential components (Fig.1):

- One or more humans controlling the mission
- A set of information about the mission and its objectives
- A data processing system

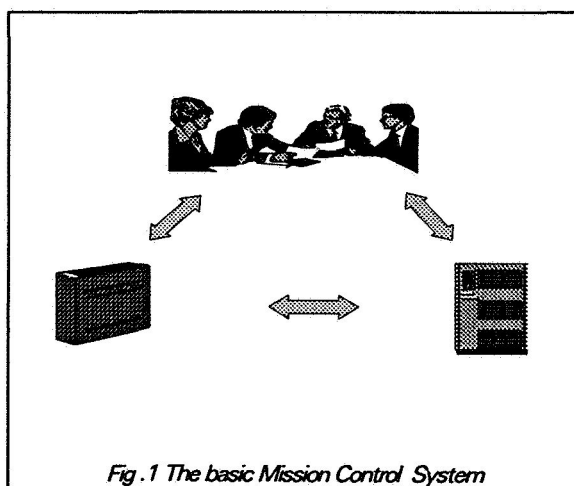


Fig.1 The basic Mission Control System

1. Mission Control Systems

1.1. Tasks of Mission Control Systems

The Mission Control System (MCS) is the interface

¹ European Space Operations Centre, Robert-Bosch-Str.5, 6100 Darmstadt, Germany

² Logica Cambridge Limited, UK

³ Logica Space and Communications, UK

In the minimum example quoted above, the mission knowledge is probably completely imbedded in the human expert's mind, and the data processing system performs a simple task of decommutating, displaying and storing telemetry, and transmitting telecommand entered by the operator. This is obviously only feasible in the case of a very simple spacecraft and mission.

The more complex our spacecraft system gets, the more involved becomes the relationship between the mission control elements shown above.

The single human expert is replaced by a team of experts, each dealing with a different aspect of the mission, and each having his own specialist knowledge. In case of round the clock operations, different experts of the same specialisation will work on the same subject, having to exchange information about the state of affairs at shift changes.

The knowledge about the mission becomes too complex and too vast to be held even by this team, and gets stored in paper specifications, plans and procedures, and databases imbedded in the data processing system.

The data processing system will then comprise a large number of inter-related software tasks, implemented on different processors, and connected by a variety of methods, ranging from inter-process communication to access to common data bases to off-line (i.e. via the human user) transfer of data. An example of the latter method, still very much in use in mission operations, is the reading from a printout of the result of a mission planning task, say the time of a particular payload action, and the manual transfer of this result to a command scheduling task.

1.2. Requirements Drivers

The most obvious reason for this ever-increasing complexity of mission control systems is that missions become more ambitious and requirements of payload users more demanding. This trend is not just linked to manned missions; in particular in unmanned satellite systems we find:

- operation of multiple payloads at the limit of the resource envelope
- numerous payload modes
- higher payload duty cycles
- shorter mission planning cycles
- direct user commanding of payloads
- shorter payload data turnaround requirements
- higher payload service availability requirements.

Increased mission requirements in turn lead to more complex spacecraft designs, which use

on-board resources more efficiently, and are manifest in:

- automatic and autonomous on-board functions
- highly optimised subsystem redundancy concepts
- extensive use of on-board software
- complexity of on-board data handling systems.

The increased complexity of spacecraft directly translates into the need for improved functionality and performance of mission control systems. However, it also means that the scope for on-board malfunctions increases and that different on-board subsystems and operating modes are much more inter-dependent. This requires qualitative increases in the

- understanding of system design
- evaluation of subsystem dependencies
- analysis of operating constraints
- operations safety measures
- contingency handling

in order to operate the mission reliably and with high availability, and to safeguard the high investment in the mission. Most of this increased complexity and functionality of the MCS will have to be absorbed in its data processing element.

However, despite the fact that missions and spacecraft become more complex, the cost of operating them is supposed to stay at a constant level or to even decrease. This of course is most apparent in the commercial space sector, but also operators of long lifetime scientific missions and operational services simultaneously operating of families of spacecraft have to respond to pressures of reducing the share of operations in the overall budget.

Since the most important cost element in mission operations is the manpower needed for preparing and executing a mission, the design of future mission control systems must address the improved use of manpower resources during all mission phases by giving the operations staff the tools enabling them to perform tasks more efficiently.

1.3. Technological Drivers

One of the most striking developments in the Information Technology industry is of course the dramatic increase in the hardware processing power, putting previous mainframe performance at the disposal of workstation users. Linked to the resulting developments in work station technology is a push towards the goal of Open Systems. Open Systems is not a well-defined term, but it will give increased flexibility to the developers of data processing systems in terms of:

- Hardware vendor independence;

- Application inter-operability;
- Applications having well-defined application interfaces so that they can be substituted with a functionally equivalent system without upheaval to other applications which use them;
- Standardisation of application programmer interfaces.

All of these features are important for the development of future MCS; firstly they have implications with respect to the nature of the tools and environments which will be provided by both hardware and software vendors and secondly they have architectural implications for future MCS in their own right.

Perhaps the most significant impact that these developments will have is that an MCS will no longer be a single software entity, but rather will be an integration of proven general purpose tools, software tools and spacecraft operations specific applications.

This is as much a necessity as a possibility since more capable mission control systems will require ever more sophisticated environments and tools to support the operator effectively. However, in order to meet financial constraints at the same time, it is essential that we are able to use commercial applications when appropriate within an MCS and that we are able to reuse software effectively from one MCS to another. An MCS will in future be a collection of integrated applications which provide an overall functionality, rather than a single, indivisible, software entity

In parallel to these developments in processing power and architectures, there have been dramatic improvements in the software discipline in terms of

- knowledge based systems
- advanced man-machine interfaces
- data base management
- software engineering

In particular knowledge based systems have evolved within a few years from a research discipline to revenue-generating systems. In space missions, the natural application for knowledge based systems is in the mission control, where they can help to analyse problems which are too complex (due to the amount of underlying knowledge) for a restricted team of operators to solve within a restricted time scale, or which are of a highly repetitive nature.

Previous ESA research projects have identified the role to be played by knowledge-based systems in future mission control systems. These projects, in the form of prototypes for specific problem domains and specific projects, have successfully demonstrated the usefulness of this technology in mission control

applications. The challenge to be addressed now is the full integration of knowledge based systems into an operational MCS.

The latest development in the progressive application of knowledge based systems is the realisation that an important cost factor of such systems lies in the collection and encoding of the knowledge required to exploit them. This has resulted in the recognition of knowledge as a 'corporate resource', and in efforts to improve the re-utilisation and sharing of knowledge between different and diverse knowledge driven data processing applications.

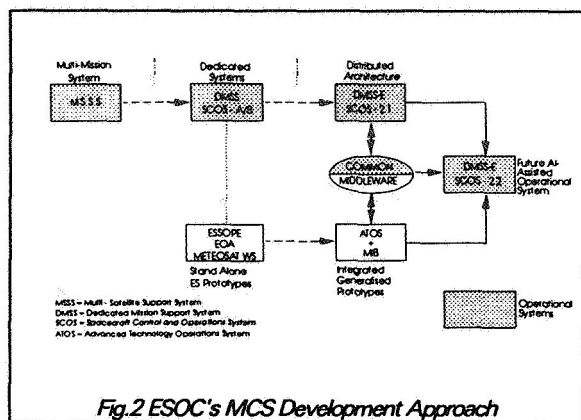
2. ATOS

2.1. Objectives

In order to exploit the available technological advances in data processing technologies, such as knowledge based systems, man-machine interfaces and data base techniques to meet the challenge for improved functionality, performance, reliability and efficiency of future mission control systems, ESA's European Space Operations Centre (ESOC) is pursuing a parallel approach (Fig.2).

ESOC's current multimission software infrastructure, SCOS-A and SCOS-B, which is based on DEXC/VAX mini computers, is planned to be succeeded by SCOS-II (Ref. 4) with a completely open systems approach based on UNIX work stations, allowing configurable, distributed processing and employing object-oriented technology.

At the same time ESOC has initiated the development work advanced mission control system capabilities under the project name ATOS, for Advanced Spacecraft Operations System. ATOS shall complement 'third generation' mission control systems such as SCOS-II as a generalised, integrated data processing platform for advanced mission control and mission support modules, employing state of the art technology with special emphasis on knowledge sharing, automatic reasoning and man-machine interfaces.



ATOS will allow to progressively add advanced control modules for use along with modules of the conventional spacecraft control system so that the advanced control functions can be evaluated under realistic operational conditions, with the conventional functions providing a safe fall-back, until for appropriate applications the advanced modules can fully take over.

Pre-ATOS prototype developments at ESOC have placed emphasis upon the use of tools and technologies in order to fabricate applications and have paid little attention to representational issues. This has led to a situation where the applications that have been built to date cannot be integrated with each other or with the classical elements of an MCS. In some cases even the tools used to build them are no longer available from the suppliers. When the tool goes so does the knowledge present within the system. Often these tools do not formally define the semantics of their knowledge representations. To make matters worse, the inferencing mechanisms often contribute to the semantic content of the knowledge. For all these reasons ATOS is placing emphasis upon the representational aspects of the system rather than the technologies that may be used to build the final architecture.

These overall objectives for ATOS are supplemented by the following generic system requirements:

Mission Independence

Efficiency in the use of manpower resources starts in the development stage for the mission control system for a new mission. Here it is important to be able to base any new development on generic systems which may be adapted to the mission in question with a minimum of additional effort. For ATOS this means the creation of generic intelligent

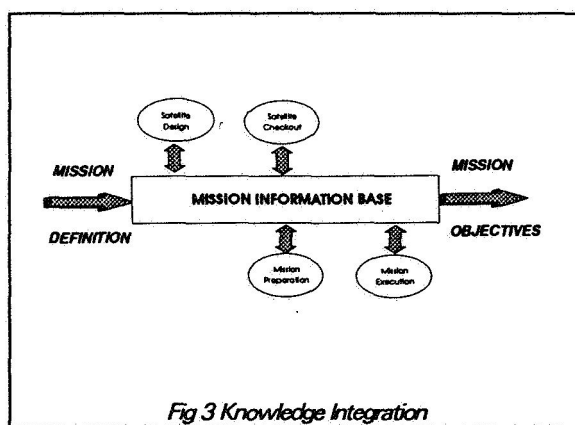
application modules for certain mission control tasks which are problem oriented, not mission oriented.

This requires the removal of all problem and mission oriented imbedded knowledge from the application code, and its storage in parametrised form outside the application so that by changing the external knowledge base alone the application may be set up for the support of a particular mission. It also requires that the representation of such external knowledge must be abstracted from the features of any particular mission.

Task and Knowledge Integration

Increased functional capabilities of the overall system, and improved efficiency in operational exploitation of the system will be achieved by the linking of different intelligent application modules of ATOS. This task integration shall allow different ATOS (and other MCS) applications to share and update external knowledge relevant to their function in a dynamical fashion. It goes without saying that this requires careful strict configuration control measures.

Applications shall also have access to different elements of the mission knowledge, meaning that logical links must be established and maintained between them, allowing the intelligent applications to use these links for enhanced depth of reasoning.



Implementation Independence

Artificial intelligence tools such as will be used in ATOS are undergoing constant evolution, and the tools which may be used in future ATOS modules cannot be predicted now. For this reason, ATOS will be designed as an open system both with respect to representation schemas, paradigms and application tools in order to be able to accommodate future developments without essential changes to the ATOS system concept.

Re-use of Knowledge

The last three requirements can be subsumed under the term re-utilisation of knowledge, to be assured

- between different missions
- between different applications and services
- between different phases and generally in time.

Growth Potential

The system design of ATOS must allow for future growth of the system in the following respects:

- accommodation of future intelligent applications not presently planned for or considered;
- improvements in computer reasoning in support of human operator tasks;
- increases in the volume of knowledge to be processed for a particular application or in mission complexity.

2.2. Architecture

Given the above requirements, the ATOS concept is defined as the combination of:

- A system architecture (including the definition of external interfaces) allowing the overall objectives to be met and guaranteeing the compatibility with the conventional spacecraft control system
- A distributed, but logically unique, repository of mission data and knowledge (i.e. information), called the Mission Information Base (MIB), and the associated tools;
- A set of advanced knowledge based system applications (Advanced Application Modules - AAM) interacting with the above knowledge and complying with a common set of internal and man-machine interfaces.

In order to practically realise such an architecture it is interesting to partition the application domains and analyse which information is needed within each domain, and particularly, where these information sets intersect.

Four generic application domains are considered to cover the functionality of mission operations:

- Operations Preparation
- Mission Planning
- Operations Execution
- Training.

The central problem is to identify the information required in support of these domains, and in particular, that information which intersects the domains, in order to permit interworking between the various

application modules. For example, planning takes place during the early phases of a mission. Not all preparation and mission design activities will have been completed. There are many interactions between these activities and the system must give support to the users in this regard. Planning, and re-planning, also occurs during the operations phase and must take account of the current state of operations, the spacecraft and its environment. If different applications are to successfully interwork and collaborate in achieving mission goals, they must agree as to which information is shared.

Although it is possible to think of the MIB as a single logically unique entity, it is unlikely that it will be realised as such. Certain partitions of the MIB will be closely concerned with the different application domains. It will be natural, and efficient, for these partitions to be located with the applications that utilise them. However, in order for the applications to interwork it is necessary to adopt some agreements on the form of the shared information. Such an agreement is called as "Ontology". In philosophy, an ontology is a theory about existence. In the artificial intelligence community the term is used to indicate definitional information concerning the "content" of a knowledge base.

Ref.2 defines the term as follows: "The ontology of a system consists of its vocabulary and a set of constraints on the way terms can be combined to model a domain."

2.3. The Operations Ontology

The establishment of an ontology for the domain of spacecraft operations is seen as a crucial component of the ATOS architecture. Without agreement on the form of shared elements of the mission information base there can be no interworking between applications. In this sense an ontology in a KBS application plays a role very similar to that of a database schema in a conventional data system. The crucial difference however is that, since we intend to develop advanced applications which reason about mission information, the form of the ontology needs to be expressed in a way which provides unambiguous semantic intent. Traditional database schemas describe the form of data, but not the content and semantics of the data. Ontology development is at the forefront of knowledge based system development.

The ontology permits ATOS applications to exchange information without loss of semantic content. That is, applications can publish information in a form in which other applications can understand.

The mission information base at any instant comprises the entirety of knowledge known to all applications. All applications have access to all knowledge at all times, although each application may (is likely to) preserve certain information in forms suitable for efficient local processing in a representation specifically designed to permit efficient manipulation in the context of one application.

For example, a planning system may require a model of the spacecraft design to be available in a representation which is efficient for the propagation of functional system modes. Such a representation may be entirely inadequate for use in a modelling application used during computer assisted operations. Rather than agree on a standard (compromise) representation, each application should be free to use whatever representation is appropriate, while at the same time being able to publish, and be notified of changes, within the shared knowledge, by virtue of agreement on ontology.

The ATOS architecture is at once efficient, open and practical. It takes account of the practical issues facing knowledge base and knowledge base application implementation, while at the same time creating, through the ontology, a resource able to live on beyond the availability of any particular tools employed in its construction.

The ontology resource, being represented in a machine readable (and translatable) format, shall be able to be utilised in the construction of schemas for various mission information base technologies and in the creation of well-formed coupling interfaces between applications and the distributed access service (Fig.4).

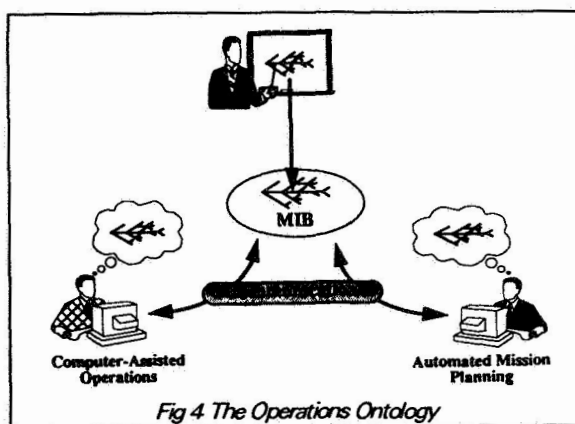


Fig 4 The Operations Ontology

2.4. Mission Information Base

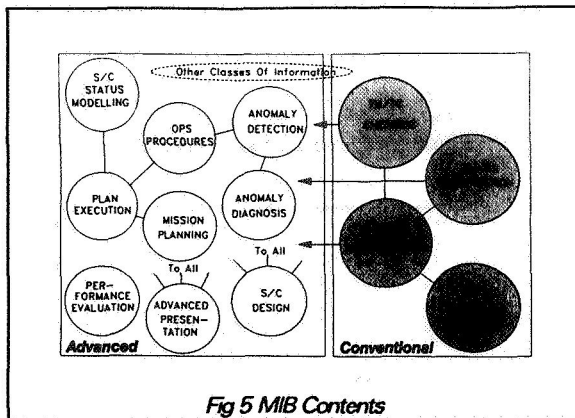
The Mission Information Base as the common repository for all system and mission knowledge has two important integrative aspects:

- It provides continuity and integration throughout the different phases of the mission life cycle:
 - Mission and satellite design
 - Satellite check-out
 - Mission preparation
 - Mission execution
- It also provides integration between the conventional and the advanced technology mission control system elements on one hand, and between the different ATOS application modules on the other hand.

These integration objectives require that the scope of the Mission Information Base contents covers information for all mission phases and for both conventional and advanced mission control modules. In terms of variability of information with time, we consider the following categories:

- Static mission information: fixed information about the spacecraft design, check out results, invariable operations procedures, parameter definitions, static knowledge bases, subsystem models
- Dynamic mission information: current spacecraft configuration, resource data, dynamic knowledge bases, dynamic operations procedures
- Transitory mission data, generated during the mission and immediately processed, distributed and then archived for future reference, i.e. housekeeping and payload telemetry and spacecraft activity (telecommand) logs.

Of these categories, the MIB will concentrate on the static and dynamic mission information as the shared mission knowledge. This is however not to be understood as a central repository to which applications read and write. The crucial difference is that in a centralised approach, the only way an application can use its own representation (which we see as an absolute necessity) is for wholesale translation of bulk information into and out of the representation used by the central store. This would be highly inefficient. Rather, we would prefer to partition the information where it is needed and use representations appropriate to the task, only agreeing on the form of shared information.



Research indicates that in collaborative endeavours such as space mission design and operations, where a large number of different disciplines interact loosely, the shared information is significantly less than the whole. Significant functional advantage can be gained by agreeing on the form of this shared information so that it can be freely interchanged.

In ATOS, it is proposed that the medium through which applications will interact will be a local and wide area network "distributed access service". Applications shall attach to the service through access points controlled by "knowledge agents". Knowledge agents shall be autonomous processes able to mediate between applications that employ different knowledge representations and different query and assertion languages.

The set of knowledge agents which comprise the service shall interact using the vocabulary of the shared ontology. The service can therefore be viewed as a "knowledge bus" into which applications can be plugged. An important feature of such an architecture is that shared information is persistently maintained either by the knowledge agents, or by attached "knowledge servers". Such servers can be considered to be applications in their own right.

The Mission Information Base will therefore contain the following building blocks:

- The distributed set of databases covering various information classes and supporting a wide range of different information structures, representational schemas and links.
- Internal management which ensures consistency and configuration control
- A set of tools for data input, browsing, editing and report generation of the MIB
- The Distributed Access Service providing retrieval and update capabilities for the ATOS Application Modules with a common API accessing 'knowledge servers'.

2.5. Knowledge Reuse

By agreeing upon the form and semantic interpretation of spacecraft operations knowledge (i.e. the operations ontology) the possibility to reuse knowledge between missions is made more tractable since generic knowledge, such as generic subsystem models, can be maintained electronically in a form which is independent of the infrastructure which is being used within a particular mission.

This opens up the opportunity for operating organisations to develop libraries of standard reusable knowledge about spacecraft and mission operations which can be used across a variety of different missions irrespective of any technological changes to the MCS infrastructure which may occur.

One of the major tasks facing a mission operations organisation in preparing for a mission is the conversion of information and knowledge supplied by the satellite manufacturer into a form suitable for use within the MCS and its supporting tools. This task is actually the subject of one of the planned ATOS advanced application modules.

The envisaged common ontology of spacecraft operations will enable a mission operations organisation to identify the information and knowledge it requires from the satellite industry irrespective of the specific software employed within the MCS. Furthermore, the ontology can be used to define knowledge capture tools which can be used by the industry to supply the required information and knowledge in an electronically readable format ready for direct incorporation into the Mission Information Base.

3. Planned Implementation and Evolution

The ATOS programme has started in 1991 and initially covers 4 individual projects until 1995. Within the programme, two distinct phases can be identified, overlapping however in time:

- The system and concept phase (ATOS-1)
- The applications development phase (ATOS-2 to ATOS-4), covering the development of pre-operational application modules for advanced mission preparation, automatic mission planning and computer assisted operations.

3.1. System and Concept Phase

The ATOS-I project is concerned with defining the overall ATOS concept and in providing the

specification of the mission information base. This involves the following tasks:

- The formal specification of an Ontology of Spacecraft Operations: This must be seen as the "first draft proposal" for such an ontology. It is expected that it shall be further develop, and validated, following ATOS-I and within the other ATOS phases.
- Translation of the ontology into the logical schema of a carefully selected mission information base foundation technology: The ontology shall be studied in ATOS-I to select the most appropriate foundation technologies to support full implementation of ATOS. Having identified candidates, the ontology shall be translated to provide the required database/information base schemas. It is likely that this will involve the integration of relational, object oriented and semantic information models.
- The architectural design of the Distributed Access Service: this shall include the design of knowledge agents able to provide services in support of application inter-operability and of application access to shared information.

The ATOS-1 project is aiming to provide as complete a mission operations ontology as possible, together with a proof of concept demonstrator. In order to be convincing, the demonstration must include the following elements: ontology translation, knowledge level interaction/reasoning between knowledge agents, together with the interface between applications to the knowledge bus, a so-called "knowledge manipulation and query language".

Several prototype elements have been identified and are planned to be integrated. An important candidate prototype consists of an environment in which the ontology can be further developed. This would permit transfer of the ontology to authorised parties within the ATOS programme for the development of specific application modules and, potentially, to allow use of the ontology outside of the ATOS programme in support of the development of advanced applications throughout the industry.

3.2. Applications Development Phase

This phase shall design and develop prototypes of certain advanced application modules (Fig.6). To do this will require access to the ontology developed in ATOS-1, together with tools to support translation of the ontology. Within the ATOS programme there will be a need for co-ordination of development of the

ontology. The draft ontology being developed in ATOS-I must be adopted, extended and developed, under proper coordination, by all other ATOS projects if the viability of the approach is to be realised and full benefit brought to the programme. Further details of the measures necessary to achieve this are described in the companion paper (Ref. 3).

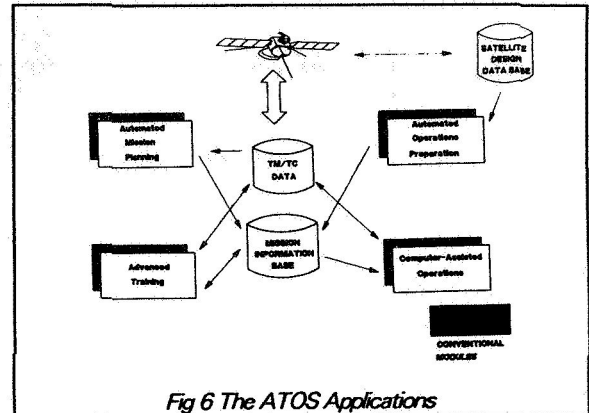


Fig 6 The ATOS Applications

Automated Mission Preparation

Presently, the most time and resource consuming task for the operations staff is the preparation of system files and operations documentation prior to their actual use in simulations and operations:

- Generation of Flight Control Procedures from manufacturer's design information, operations handbook(s), subsystem specifications;
- Generation of telemetry and telecommand parameter characteristics files for use by the spacecraft control software, from paper satellite specifications or from a satellite data base also used for checkout purposes;
- Generation of auxiliary spacecraft control information in files or in code, such as mode equations, derived parameter definition, high level command sequences, mission constraints, based on the results of the two activities above, and on additional spacecraft and mission information.

This data is subject to frequent updates during the preparation period and sometimes even after launch, due to design changes, and test and actual operations results. In all these cases, the effects of an update in one item on the rest of the data has to be carefully checked and considered.

The automated operations preparation module will support the operations engineer in this task by

automatically transcribing satellite design, build and test data into the appropriate representations and by:

- Reduction of high volume input data from the satellite design and development process, existing in diverse formats, to a defined structure;
- Evaluation of dependencies in the input data and generation of logical links and conditions in the mission information base, using automatic reasoning based on the mission operations ontology;
- Automated Validation of generated Flight Control Procedures for completeness, consistency and operational safety;
- Efficient high level interface with the human operations engineer who remains responsible for conflict resolution and final vetting of the mission information.

Automated Mission Planning

Automated mission planning addresses the automated generation of optimised plans for payload and spacecraft exploitation and configuration, astronaut activity planning and ground facilities scheduling.

The resulting plans shall maximise mission output (activities, products), while minimising the use of depletable or expensive resources (e.g. power, fuel), and satisfying a set of constraints (e.g. safety, deadlines, environment, available resources).

This covers initial mission planning well in advance of actual mission operations as well as near real time re-planning necessary to optimally react to unforeseen changes in the execution of the plan, or in the environment.

Computer-Assisted Operations Execution

Computer-assisted operations address the core mission operations tasks:

- Re-configuration of spacecraft and payload in execution of agreed mission plans or as required by the space environment;
- Manoeuvring of spacecraft attitude and orbit;
- Continuous health monitoring of spacecraft and payload;
- Contingency actions in case of deviations from nominal satellite state and operations.

These tasks shall be covered by a number of intelligent support functions for the mission operator:

- Automatic generation of command sequences for complex Flight Operations Procedures

including the necessary feedback from the satellite;

- Automatic telemetry monitoring for comparison with nominal state and trend analysis of critical parameters;
- Automatic direct and indirect anomaly detection and diagnostic support;
- Generation of contingency command sequences based on intelligent failure diagnostics and partially pre-defined procedures.

Adaptive Training

Safe and efficient conduct of mission operations requires comprehensive initial training and repetitive proficiency training of staff throughout the operational lifetime of the mission. This is of particular importance where, due to the overall extent of the operations, tasks become fragmented, and where a long mission duration prevents team continuity. Training can be thought of as a controlled simulation of nominal and contingency operational situations requiring adequate response from the trainee, who is presented with an interface close to the one with a real satellite. The adaptive training module, planned for later implementation within the ATOS environment, shall allow to exercise mission operations using the results of the other ATOS modules adapted to the trainee's tasks, using the knowledge about the mission stored in the MIB.

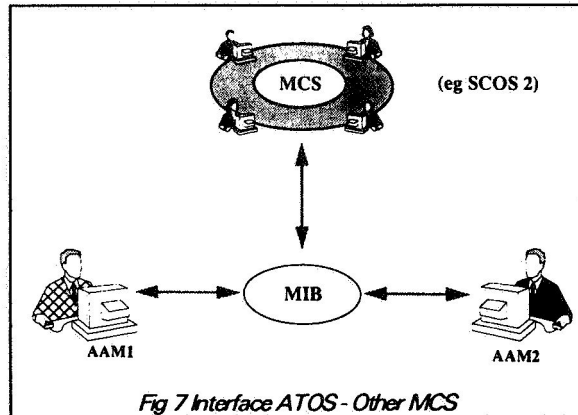
3.3. Integration with conventional MCS

As stated above, ATOS is being developed primarily to complement ESOC's new mission control system SCOS-II, but the ATOS concept should of course be usable to enhance the capabilities of any reasonably advanced MCS. The foreseen interaction between ATOS and SCOS-II will be a good model of the interaction of other MCS with ATOS.

The fundamental link between the basic MCS and the ATOS applications will be constituted by the MIB, which will be based on the common mission operations ontology. The MIB for a particular mission will logically be considered as a single logical entity, but physically it may be implemented as multiple entities with different storage mechanisms. The MIB can be accessed by messages through the Distributed Access Service passing knowledge requests, which will be served by knowledge servers attached to the MIB or even by intelligent ATOS applications. Such requests may come from ATOS applications or from application of the basic MCS. In fact, in order to avoid overlaps in functions, ATOS applications may make

requests through the MIB DAS for basic MCS services (such as telemetry parameter limit checking) to be handled by the basic applications.

Through these mechanisms the scope and functionality of the basic MCS is widened by the advanced ATOS applications, and all applications of this wider MCS are of equal status and can interwork closely through the MIB DAS.



4. Conclusion

In as much as it is aiming to advance the state of the art with respect to computer reasoning (intentionally avoiding the term artificial intelligence) in space mission control systems, ATOS must certainly be regarded as an ambitious project. Yet the real ambition lies not so much in the exploitation of the latest data processing technology and knowledge based systems, but in the objective to create a generalised framework which will be open to future advances in the field, and will remain valid beyond the lifetime of currently available algorithms and tools.

The other novel element addressed by ATOS is the concept of operations knowledge sharing and re-use across different dimensions, i.e. lifetime, applications and organisations. The key to this is agreement on the mission operations ontology, which will provide the common understanding on the interpretation of models of the operations domain.

The validity and even necessity of advancing the re-use of knowledge is of course not limited to the problem domain of space mission operations. It has been recognised in many application areas that the high cost of building new information processing systems for the mastering of complex entities requires the re-use of knowledge in the same way that standard software tools are already re-used in present developments.

In the mission operations domain it is probably inevitable that different mission operations organisations, because of differences in their operational requirements and because of different traditions, will continue to develop differing MCS infrastructure tools. However, by agreeing to standardise upon a common ontology of spacecraft operations, in those areas where it is required for the tools to inter-operate, we can establish a practical mechanism for exchanging and sharing knowledge between them.

References

1. M. Curkosky, R. Engelmores, R. Fikes, T. Gruber, M. Genesereth, W. Mark, J. Tenenbaum, J. Weber, "PACT: An Experiment in Integrating Concurrent Engineering Systems", Submitted to IEEE Computer January 1993 (Special Issue in Computer Supported Concurrent Engineering).
2. R. Neches et al, "Enabling Technology for Knowledge Sharing, AI Magazine", Fall 1992.
3. K.J. Poulter, H.N. Smith "ATOS-1: Designing the Infrastructure for an Advanced Spacecraft Operations System", Proc. 2nd International Symp. "Ground Data Systems for Space Mission Operations", SPACEOPS '92, 17-20 Nov. 1992, Pasadena.
4. N. Head, J.F. Kaufeler " Evolution of the Agency's Software Infrastructure for Spacecraft and Mission Control", ESA Bulletin No. 67, August 1991

N94-23898

NASDA Knowledge-based Network Planning System

K.Yamaya¹, M.Fujiwara¹, S.Kosugi¹, M.Yambe², M.Ohmori³¹Tracking and Control Center /
National Space Development Agency of Japan²Fujitsu Limited, Tokyo, Japan³Fujitsu Social Systems Engineering Limited, Tokyo, Japan

ABSTRACT

One of the SODS (space operation and data system) sub-systems, NP (network planning) was the first expert system used by NASDA (national space development agency of Japan) for tracking and control of satellite. The major responsibilities of the NP system are: first, the allocation of network and satellite control resources and, second, the generation of the network operation plan data (NOP) used in automated control of the stations and control center facilities. Up to now, the first task, of network resource scheduling, has been done by network operators. NP system automatically generates schedules using its knowledge base, which contains information on satellite orbits, station availability, which computer is dedicated to which satellite, and how many stations must be available for a particular satellite pass or a certain time period. This paper introduces the NP system.

Key words: NP system, knowledge base, flexibility, conflict solving, expert systems

1. INTRODUCTION

This paper introduces the NASDA network planning (NP) system. Since NP system is a Type-1 SODS sub-system we start with a brief explanation of SODS.

1.1 Development of SODS

The type-1 SODS, which is the ground-station-based tracking and control system, has been in partial operation since the launch of JERS-1 (Japan earth resources satellite-1) in 1992. NASDA started to develop a SODS long-ranged tracking and control system, in 1987.

The type-1 SODS is the first step in the development. In the early stages, we expected two new requirements for satellite operation. One is for simultaneous operation of more than three sun-synchronous sub-recurrent polar satellites (JERS-1, MOS-1: marine observation satellite-1 and MOS-1b), MOS-1 and MOS-1b were launched in 1987 and 1990. The other requirement is for double launch operations for use with the H-II rocket to go into operation in 1995. NASDA automated ground network control to meet these needs. NASDA also automated the scheduling of the ground network and planning file transmission.

1.2 Network Planning System

The NP system allocates network and satellite control resources for each satellite. NP then generates data for the network operation plan (NOP) used in the remote control of the stations and in configuration of control center facilities. The NOP flow chart is shown in Fig.1. NOP data, which based on the operation requests sent from network users through the NM (network management system) referring to the station event data (such as visibility information and periods when operation is under constraints), is transferred to TC (tracking network control system) and LC (communication line control system). This data is used by TC for the station control and by LC for the communications line control.

1.3 Developing NP policies

We considered several requirements when developing the NP system. One was the store and inheritance of know-how on network planning from experts in the field. More complicated and more frequent operations make it more difficult to store and inherit the

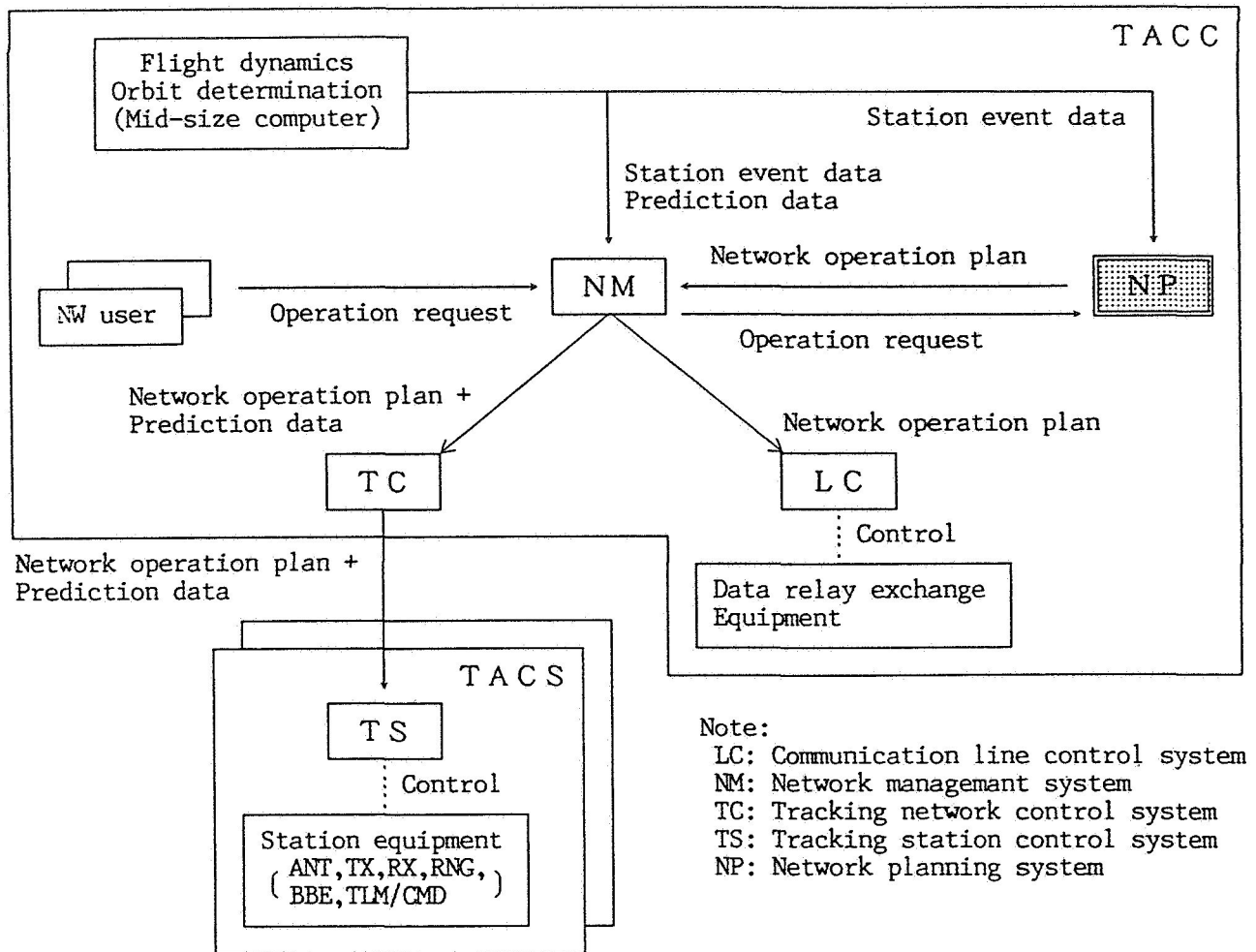


Fig.1 Data flow for the network operation plan

know-how. Another consideration was that scheduling could be simulated in some situations. Combining these factors, we introduced an expert system to NP.

1.4 Computer Technology

Type-1 SODS used a variety of new equipments, including engineering work stations (EWS), and a local area network (LAN). The NP system is the first expert system introduced in NASDA's tracking and control operations. Recently, expert systems have proven capable of using their inference capability for judgement and reducing part of the workload of an operator. Generally, expert systems can be classified into four groups: diagnostic, for medical patients or machine troubles; forecasting, for financial

affairs analysis; control, for factory automation or fuzzy systems; and design and planning, for electric circuit design, construction of computer systems, and planning travel courses. The configuration of the NP system corresponds to the planning type.

2. FUNCTIONS OF THE NETWORK PLANNING SYSTEM

2.1 Knowledge Base

The NP system relies on a knowledge base data constructed from the know-how of human experts. We divide this knowledge base into 'definitions' and 'tables'.

2.1.1 Definitions

Definitions are of resources (satellites

in operation, network facilities, etc.), operation priorities, and satellite priorities. The NP system uses these definitions for scheduling. The operator must define the resources to be used before scheduling an operation. Definitions contain the following information.

(1) Priority

The NP system's knowledge base uses two forms of priority: operation (launch, maneuver, usual, ...) and satellite. Each plan is established in the order of operation priority. When operation priorities are equal, each plan is established according to satellite priority.

(2) Satellite

Information is included for each satellite (ID, name, type of orbit, time margin of equipment preparations). This Definition is important for scheduling. The NP system changes the scheduling method in accordance with satellite orbits. Operation time is also decided by the required time for the station facility set-up.

(3) TACS (tracking and control station)

TACS information containing TACS name, TACS ID, and TACS type (domestic or foreign).

(4) TACS Antenna

TACS antenna definitions including antenna name and antenna type (X-Y or AZ-EL).

(5) TACC (tracking and control center) facilities

Contains TACC facility name and ID.

(6) SOCS (satellite operation and control system)

Includes SOCS name and ID.

(7) Station/pass combination definition

For sub-recurrent polar orbit. Defines station/pass combination pattern in detail. Some examples of this pattern are shown in Fig.2. More patterns are used in actual operation.

2.1.2 Tables

Tables indicate how to use resources, which computer is dedicated to which satellite, and how many stations must be available for a particular satellite pass or a certain time period. Each table has 'flexibility' for an optimal schedule where flexibility is expressed as the preferred order of facility allocation which are defined above. Using flexibility, when a facility

ex.1) 2 pass operation

	Pass 1	Pass 2	Pass 3
Station A	■	■	

ex.2) 2 station / 2 pass operation

	Pass 1	Pass 2	Pass 3
Station A	■	■	
Station B	■		

ex.3) 2 station / 3 pass operation

	Pass 1	Pass 2	Pass 3
Station A	■	■	■
Station B		■	

Fig.2 Station/pass combinations

cannot be used due to conflicts, alternate facilities are assigned. A table is constructed for each satellite. Tables are constructed for the following information.

(1) Preferred operation day

Prioritizes scheduling days for geostationary orbit.

(2) Preferred operation time

Prioritizes operation times for geostationary orbit.

(3) Preferred ranging time

Prioritizes ranging time. One ranging unit is a quarter for each one hour, for geostationary orbit.

(4) Preferred facilities

Gives TACS antenna preferences for geostationary orbit.

(5) Preferred SOCS

Prioritizes SOCS for geostationary and sub-recurrent polar orbits.

(6) Station/pass combination & activities

Prioritizes 'station/pass combinations definition' for sub-recurrent polar orbits.

(7) Sub-recurrent orbit nominal pass

Defines TACS antenna and pass preferences, where pass information fits in to visible patterns for sub-recurrent polar orbit satellites.

(8) Request code table (executive parameters & activities)

This table serves two purposes: user services and facilitating the knowledge base. Users can request satellite

operations simply by using numbers from this table. Users are not required to choose the operation time or revolution number, these are established from other tables. This table also provides information for the knowledge base. When a request and other requests conflict, optimal plans are determined by this table.

The request code table contains detailed request parameters and numbers from some of the above tables concerning request flexibility. The two orbit types considered are geostationary and sub-recurrent polar orbit. Table combinations are shown in Tab.1 and Tab.2. Since many table ID combinations are possible, numerous request code table preferences are possible.

2.2 Scheduling By Network Planning System

As mentioned, the NP system establishes plans for knowledge database use. Requests are sorted by operation and satellite priority. NP system has three different planning modes for a variety of approaches for solving request conflicts.

2.2.1 Scheduling Modes

The NP system has three planning modes.

(1) Rejection mode

When a request conflicts with an already established plan, the request is just rejected. The request is regarded as 'NG,' and is unavailable for operations.

(2) Solving selection mode

When a conflict occurs, operators can select the conflict solving method. The methods are 'cut & try,' 'priority exchange,' 'backup code' and 'rejection.' These methods are explained in 2.2.2. Method selection allows efficient planning and can produce a tight fitted schedule.

(3) Automatic solving mode

Unlike the solving selection mode, this mode needs no operator intervention. Conflicts are solved by a fixed procedure. First, cut & try is tried. Second, if a request becomes 'NG' by cut & try, backup code is attempted. Lastly, if both cut & try and backup code fail, the request is put into rejection mode.

The solving selection and automatic solving modes make the most of the flexibilities for solving request conflicts, and for improving schedules.

Tab.1 Geostationary orbit

Request code table	—	Preferred operation day table
	—	Preferred operation time table
	—	Preferred ranging time table
	—	Preferred facilities table
	—	Preferred SOCS table
Form of request (lesser items)		Examples
Term of operation	...	Nov. 12 to Nov. 20
Request code ID	...	01 (2 TACS RNG operation)
Operation class (referring to operation priority)	...	04 (Usual operation)
Backup operation code ID (Cf. 2.2.2 conflict solving)	...	03 (1 TACS operation)

Tab.2 Sub-recurrent polar orbit

Request code table	—	Preferred SOCS table
	—	Station/pass combination & activities table —
	—	Station/pass combination definition
	—	Sub-recurrent orbit nominal pass table
Form of request (lesser items)		Examples
Day of operation	...	Nov. 12
Request code ID	...	01 (3 TACS operation)
Operation class (referring to operation priority)	...	02 (Maneuver operation)
Backup operation code ID (Cf. 2.2.2 conflict solving)	...	05 (2 TACS operation)

2.2.2 Conflict Solving Methods

This section explains the conflict solving methods used by the solving selection and automatic solving modes.

(1) Cut & try process

We assume NP is processing "Request X." NP seeks available network facilities and operation time within the flexibility of "Request X." Conflict then occurs.

① The group of already scheduled requests which conflict with "Request X" are cancelled.

② "Request X" is scheduled.

③ Available network facilities and operation times within the flexibilities of the cancelled requests are searched for.

④ If cancelled requests cannot be scheduled, NP cancels "Request X" and recovers the cancelled requests.

(2) Priority exchange process

The difference between cut & try and priority exchange is how they are done if a cancelled request cannot be rescheduled (as in ④ below). By priority exchange, cancelled requests become 'NG.'

① Already scheduled requests which conflict with "Request X" are cancelled.

② "Request X" is scheduled.

③ Available network facilities and operation times within the flexibilities of the cancelled requests are searched for.

④ If the cancelled requests cannot be scheduled, NP does not recover the cancelled requests. "Request X" plans are left.

(3) Backup code

If a request cannot be scheduled, the NP system schedules using this backup code. For example, we suppose that the usual code designates 3 TACS operations and backup code designates 2 TACS operations. If 3 TACS cannot be established, then 2 TACS are scheduled. Of course, users are not always required to use the backup code.

(4) Rejection

Requests are rejected.

2.3 Environment

The NP system runs on an engineering work station (Fujitsu A-80). The operating system is UNIX (SX/A) and the principally used languages are C, LISP, and FORTRAN. The NP environment is shown

in Fig.3.

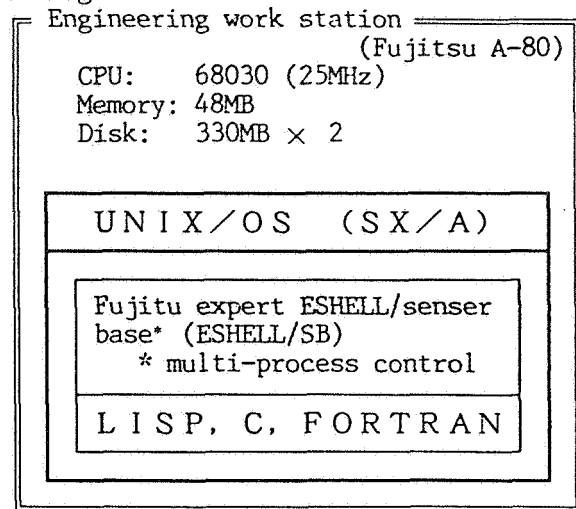


Fig.3 Operating environment

3. APPLICATION TO OPERATION

NP operates according to the timeline of Fig.4

(1) Monthly scheduling period

In this phase, the operator schedules activities such as facility maintenance, launches, and the operation plans for each satellite's nominal orbit. This schedule is a base of weekly and emergency scheduling. Monthly network schedule lists are printed out.

(2) Weekly scheduling period

In this phase, the operator reconsider the monthly schedule in more detail for the given week. If problems occur, scheduling is executed again. Weekly network schedule lists are printed out.

(3) Active scheduling period

If a problem happens during this phase, the operator updates the schedules as emergency scheduling. A daily network schedule lists is printed out.

4. CONCLUSION

4.1 Evaluation of NP System Operation

The NP system may be applied to a maximum of 28 satellites and 12 antennas with up to 99 operation codes per satellite. Results for scheduling three sub-recurrent polar orbit satellites were satisfactory. Before the launch of JERS-1, there were only two polar satellites whose visible time had very little overlap. However, the visible time of JERS-1 overlaps theirs so

N94-23899

A CROSS-DISCIPLINARY RESPONSE TO IMPROVE TEST ACTIVITIES

The Corporate Memory Capitalization in Ariane4 Test Domain

Dinh Phuoc VO and Christian Soler
 Aramiihs-Matra Marconi Space
 Toulouse, France

N. Aussenac and D. Macchion
 Aramiihs-Irit

ABSTRACT

The AIT/AIV of the Ariane4 Vehicle Equipment Bay has been held at Matra Marconi Space (MMS) site of Toulouse for several years. For this activity, incident interpretation necessitates a great deal of different knowledge. When complex faults occur, particularly those appearing during overall control tests, experts of various domains (EGSE, software, on-board equipment) have to join for investigation sessions. Thus, an assistance tool for the identification of faulty equipment will improve the efficiency of diagnosis and the overall productivity of test activities.

As a solution, the Aramiihs⁽¹⁾ laboratory proposed considering the opportunity of a knowledge based system intended to assist the tester in diagnosis. This knowledge based system is, in fact, a short-term achievement of a long-term goal which is the capitalization of corporate memory in the Ariane4 test domain. Aramiihs is a research unit where engineers from MMS and researchers from the Irit⁽²⁾-CNRS⁽³⁾ cooperate on problems concerning new types of ma-system interaction.

KEYWORDS

Corporate Memory, Ariane4 Test, Diagnosis, Knowledge Acquisition, Expert System, Case Based Reasoning

I. INTRODUCTION

Information technology can help companies achieve higher productivity and better reliability in decision making, communication and activities

organization. With the large range of technical possibilities at hand in realizing applications, people become more and more aware of the importance of matching the right technique with the right need and of evaluating the impact of this solution upon organizations.

So far, we are taking part in the emerging awareness of two fundamental aspects when new information processing systems are to be defined. These features have been recently taken into account in information technology and data processing system design and development.

The first aspect stresses the evaluation of the aim impact (need analysis and organization study) and the choice of an appropriate combination of techniques to achieve this aim. The second point focuses on the time required for the aim to be achieved. If realizing the application is to take a long time, a way to turn such a long-term project (which can make end users discouraged and unmotivated) into efficient short-term ones is to propose an incremental design and development. The objective then becomes the gradual realization of an application

(1) Action de Recherche et Application Matra Irit en Interaction Homme Systeme

(2) Insitut de Recherche en Informatique de Toulouse

(3) Centre National de la Recherche Scientifique

that gives benefits to its end users at each step of its achievement and thus will maintain the interest and participation of those end users.

In this paper, we describe the development of a knowledge based system for diagnosis assistance. It is intended to be used in the context of Ariane4 Vehicle Equipment Bay (VEB) Assembly, Integration, Test and Validation (AIT/AIV) activities. This knowledge based system is, in fact, the first short-term achievement. The long-term goal is to capitalize technical memory in the Ariane4 test domain.

The long-term aim is to make knowledge well provided among people in a work environment and to have them participate in the enrichment and standardization of it. Thus, we have to plan the gradual development of the final system. The amount of the capitalized knowledge will be more important than the one needed by a 'standard' expert system. It is possible that the final system will consist of several systems (expert systems, intelligent data bases with multimedia accesses, intelligent tutoring systems, etc.). The final impact is of great importance since it both modifies the way people work (improves the productivity and skill of the staff) and the view they have of their work.

We will also detail in the paper our approach which is a cross-disciplinary one since it includes software techniques, Artificial Intelligence, and human factors evaluation. Such an approach is possible, thanks to the context provided by the ARAMIIHS laboratory, where engineers and researchers cooperate on problems raised by the development of complex man/machine interfaces.

OPERATIONAL NEEDS AND CONTEXT

2.1 The Ariane4 VEB

The AIT/AIV activities of the Ariane4 launcher VEB have been performed at the Matra Marconi Space (MMS) site of Toulouse for more than 10 years, and will continue for 10 more years before being totally replaced by the Ariane5 launcher.

The Ariane4 VEB is mechanically interdependent with the third stage of the launcher during the flight. It ensures the interface between the structure of the launcher and the satellite contained in the fuse cap.

The functionalities of the VEB allow it to:

- guide and control the launcher during the flight,
- send orders to the three stages of the launcher for the trigger or stop of the propellant systems, the separation of the stages, the release of the fuse cap and the satellite, and the change of data format of the telemetry,
- verify the state of the satellite and control its release,
- satisfy security constraints by localizing the launcher and, if necessary, by destroying it,
- allow, during the preparation of the launcher and its flight, the transmission of telemetries to control centers for the verification of the correct operation before launch and during flight,
- and process, before launch, the telecommands necessary to control and bring into operation different elements of the launcher. In figure 1, we have the different functionalities of the VEB.

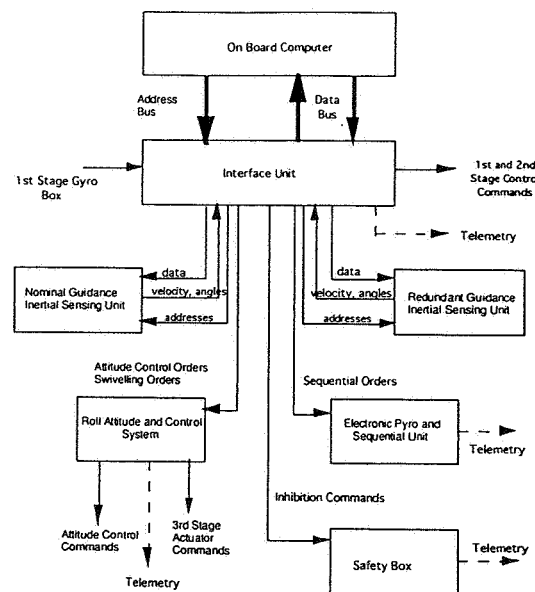


Figure 1 VEB FUNCTIONAL SYNOPSIS

2.2 The AIT/AIV Activities

As it was said previously, the MMS site of Toulouse is in charge of the AIT/AIV activities of the VEB. Those activities include the last tests on the VEB before it is sent to Kourou (the launch site).

2.2.1 Tester Behavior Confronted With an Incident

During the test phase, when an incident occurs, integration responsables begin the diagnosis by an analysis phase. This analysis phase requires understanding the context of anomalies through the appearance of symptoms, and having the appropriate knowledge, experience and information related to the incident. From this analysis phase, a hypothesis will be made. Then, there will be decisions on complementary tests to run in order to validate the hypothesis and to localize better the cause of the incident.

Thus, those activities necessitate from testers a good amount of knowledge in test environment which is composed of:

- the VEB under test
- the test bench
- the software of the on-board and ground computers
- the specification of tests

The knowledge lies both in documents and in the expert testers' experience. The documents needed during a diagnosis are numerous: ground software specifications, test specifications, test bench and VEB documents, electrical plans, etc. Testers' experience necessary for the investigation must cover all the equipment involved in the test. Part of the equipment is shown in figure 1.

Hence, when complex faults occur, particularly those appearing during overall control tests, experts of various domains (EGSE, software, on-board equipment) have to conduct investigation sessions and search for the faulty equipment. These sessions, though necessary, require important time expense from different specialists. Moreover, testers sometimes spend a large amount of

time dealing with problems that could be easily solved by experts. Thus, an assistance tool for the identification of faulty equipment will improve the efficiency of diagnosis and the overall productivity of test activities.

2.2.2 The Existing Management of Incidents

All the incidents are written down on nonconformance reports. These reports can be, if necessary, consulted to find out whether the present incident already occurred in the past. They are attached to documents describing the diagnosis approach that was taken in the search of the cause of incidents. They can help testers to elaborate an appropriate diagnosis approach for a current incident.

2.3 THE NEEDS

2.3.1. An assistance for the Evaluation of the Context

When confronted with an incident, integration responsables have to take into account the global context in which the incident appears. This is a critical phase since from this context evaluation, testers will take actions such as complementary tests to determine the cause of anomalies. A good evaluation of context will allow carrying out appropriate actions, the impact of which will not disrupt the diagnosis process and the incident context.

2.3.2 A Reduction in the Time Required for Diagnosis

When an anomaly occurs during the test phase, the rapidity of its resolution will improve the global productivity of test activities.

This is particularly true when the VEB is under tests with special conditions such as vibrations and thermal constraints. Indeed, in this context, the faster the correct diagnosis, the better it is, since that prevents exposing the VEB too long under environmental conditions that are far harder than real ones.

2.3.3 A Search for Appropriate Information to Make a Reliable Diagnosis

The diagnosis must be reliable. The performance of a reliable diagnosis depends on the identification of correct information. Hence, it is necessary, when an anomaly appears, to look for useful complementary information. That information is linked to the context of the anomaly appearance.

Another consideration is that several types of anomalies can have similar symptoms on the test results. Hence, complementary tests will permit obtaining more information. The diagnosis approach must follow a strategy which ensures an optimization of actions to be performed and a maximum of reliability and safety.

2.3.4 Adaptation of Old Incident Solutions to Present Anomalies

The rapidity, the efficiency, and the analysis approach are often achieved owing to the experience of testers. This experience relies on adapting diagnosis process from old cases to present problems. Thus, old cases can be of great help in understanding the present context and elaboration of the diagnosis approach.

3 THE RESPONSE

As a solution, we consider the opportunity of a knowledge based system intended to assist testers in diagnosis. Given an incident, the system will assist users in making good hypotheses and in having an appropriate process to run complementary test actions to validate the hypothesis.

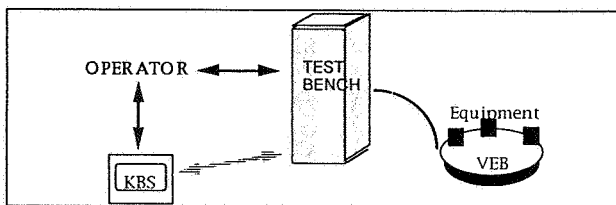


Figure 2: The integration of the expert system in the test environment

To manage the acceptability and the introduction of AI techniques and culture in this operational environment, we benefit from an expert completely involved in the development of the expert system itself. This aspect will also serve

our long term goal, which will be detailed in the next paragraphs.

The main originality of our approach is to consider the solution and the knowledge based system as a cooperative tool that should help the operator. This point of view differs from the usual ways of considering AI applications, which often neglect the user.

This focus on the user induces implicit yet significant changes in the status of the system to be developed:

- building an expert system results in putting expertise at the user's disposal through an artifact,
- the design of the system not only focuses on the expertise itself, but also on the way the expertise is communicated and made available to nonexperts,
- the knowledge based system (KBS) becomes much more dependent on its use context and on the technical environment it will be integrated in.

A KBS will not meet these requirements unless it behaves like an intelligent team-mate, a cooperative tool that contributes to the improvement of the user's task achievement with its own competencies.

Considering a KBS from this perspective has several consequences on the way it must be designed. We intend to take such constraints into account in the life cycle we follow to develop AI applications. Our methodology combines results from human factors, knowledge engineering and modelling to ensure the system acceptability by the end users.

--The work organization in which the system will be settled is to be taken into account as soon and as closely as possible. This refers to the work organization, the people concerned with the project, existing and future

tools, etc. The purpose is to manage a good integration of the system within this context. The previous paragraphs describe the AIT/AIV context and work activities.

--Building a cooperative system requires a strong reference to the user's abilities and know-how in performing the test, both with and without the help of a system. The user's task analysis provides information about the user's needs and the expected functionalities and roles of the target system. It helps specify the kinds of cooperation expected. AIT/AIV operators expect the diagnosis expert system to save time in localizing causes of defaults and to explain its diagnosis process.

--The acquisition of the knowledge required by such a system should not be restricted to the early stages of the system life-cycle, but it should be continued to enrich the system as long as it remains in use. We promote a modelling approach, where knowledge acquisition consists of building an expertise model independent of the symbolic representation in the knowledge base. This model serves as a medium of communication between the expert, the knowledge engineer and, later, the maintenance engineer. It provides a frame that guides the acquisition and modelling of new knowledge. Such a model is to be translated in order to obtain the target system. The representation primitives should help design a cooperative system, that is to say, they should make it possible to explicate how the solving process can be adapted, according to strategies or context dependent constraints.

- Validation of a KBS results from much more than the knowledge base qualification. The life cycle includes a step-by-step validation by users and experts. We consider the system as valid only after three kinds of properties have been checked: (i) it performs correctly the task it is intended to, (ii) the user understands what the system does (there is a "cognitive compatibility" between the users and the system solving processes), and (iii) the system fits in its use environment (it is integrated with existing systems, practices and tools).

In order to ensure the system usability, an experiment has been carried out with expert and non-expert operators. The language they use and the level of abstraction at which they solve a problem have been compared. The differences stressed will be taken into account in order to adapt the system interface and the kind of explanations it provides.

3.1 THE SHORT TERM GOAL

The knowledge contained in the expert system will allow dealing with the more usual problems encountered during overall control tests. Diagnosis during this particular type of test is difficult since all the equipment of the VEB is simultaneously functioning. Thus, when an anomaly appears, a global knowledge of the VEB is necessary to decide on the appropriate approach. In agreement with the expert, we decided that the system will mainly consider the cause of anomalies as unique.

At the present time, we are developing the prototype. It is planned to be delivered to testers in March, 1993. Having the system at hand will allow testers to suggest some modification or add-ons about the functionalities of the system. Presently there are suggestions from testers to implement explanation modules in the system. This will enable the system to justify and explain the deduction of its hypothesis and its diagnosis process.

In the same way, the knowledge stored in the system will be available for testers to use and to make suggestions for its enrichment.

3.2 The Long-Term Objective

Our final objective is to make VEB test knowledge available to all testers. The knowledge stored in the expert system will then be enriched by different experts of different equipment of the VEB. In effect, initially the expertise stored in our system only concerns that of an expert in VEB tests. Then, as the system is used and AI techniques and culture is better understood by end users, we will better identify which

kind of knowledge from experts of specific VEB equipment can be added. Moreover, these experts will feel much more involved in the system design at that time.

Figure 3 illustrates our approach.

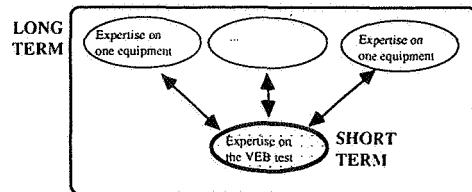


Figure 3: The expertise domain covered by the system in short term and long term perspectives

Presently, experts of different domains in integration plan to evaluate the expertise of the system at the end of this year. Thus, it will be the first session where there will be a discussion on the quality of the system knowledge and also proposals to enrich it.

Thus, by means of the system, we plan to make the knowledge as living as possible in the growth both of quality and quantity. And as the knowledge grows, the functionality of the system will grow as well. The results may be developments of different applications. Indeed, as we know better the context of test integration environment, we will have a deeper insight as to what can be done to still improve test activities in quality and productivity.

To conclude, we aim at capitalizing the corporate memory of the test environment. This task implies gradual achievements which result in the development of advanced information processing systems such as expert systems. The making of these systems in turn will encourage people in test environments to improve and enrich the corporate knowledge in the Ariane4 test context.

4 THE APPROACH

To achieve the short-term goal we are carrying out the knowledge acquisition, the definition and realization of the system architecture and the definition of the man/machine cooperative model.

The first two phases are more complex

to achieve than by simply operating in the context of realizing a simple knowledge based system. Indeed, we have to lay clues for the long term goal. The last phase is useful for us to take into account the different constraints that are applied in test environments and, by this, to study the behavior of testers in various situations and contexts. This will help us to design well-fitted systems to end-users needs.

4.1 Knowledge Acquisition

As mentioned earlier, we stress the interest of knowledge modelling for knowledge acquisition. The model facilitates the understanding of the knowledge stored in the system and of its behavior when performing a task. It makes explicit the way the system will solve a problem, adapt its behavior according to the current goal and context, etc. The conceptual model can be considered a detailed specification of the knowledge base, independently of how it will be implemented.

Such a model is used, of course, to acquire knowledge from the expert and to build up the system, but also to maintain it, to generate better explanations, etc. Even though making it explicit takes time, it turns the famous "knowledge acquisition bottle-neck" into a succession of modelling activities much more guided and efficient than rapid prototyping.

In this project, knowledge acquisition is carried out following the MACAO methodology and using the associated tool. This method promotes a bottom-up modelling process from the analysis of cases solved by the expert. Model-based knowledge acquisition can be decomposed in four major stages:

- data-driven knowledge acquisition where information (rough data) is drawn from the various knowledge sources (expert's activity, documents, etc.) without any predefined idea about the expertise,
- abstraction of a schema of the conceptual model: the tasks performed by the expert are characterized as well as the methods he uses and his strategies; also his

representation of the domain entities is associated to a kind of model structure,

- building of the full conceptual model: the schema of the model is filled with all the required application knowledge until the model is complete; the schema may be consequently refined,

- implementation of the conceptual model as the target system knowledge base.

Figure 4.1. shows the different stages in knowledge acquisition and modelling.

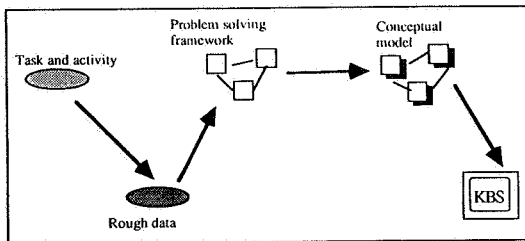


Figure 4.1.: Stages in knowledge acquisition and modelling

With MACAO, several techniques are suggested in order to carry out each stage. A software helps build up the model: it proposes knowledge browsers and editors, structures for the model formalization. The conceptual model consists of two components: a domain model which gathers static knowledge about the expertise domain (concepts and their relations); a problem solving model that represents the task decomposition describing how a problem can be solved. Each model is represented with a specific language: semantic networks for the domain model, a graph of schemas for the problem solving model. Both are shown through graphs editors.

The techniques promoted by MACAO are derived from human factors psychology or ergonomics, as well as knowledge acquisition. For instance, for data-driven knowledge acquisition, one of the techniques proposed is problem simulations, followed by focused interviews bearing on explanation of the simulations. Repertory grids are used to identify classes of problems. The problems solved by the expert

form a set of cases that are compared in order to abstract the characteristics of the expertise and the schema of the conceptual model.

The graph in figure 4.2. represents the schema of the conceptual model obtained with our application. The application is solved with a method close to systematic diagnosis, and exploits a functional model of the V.E.B.

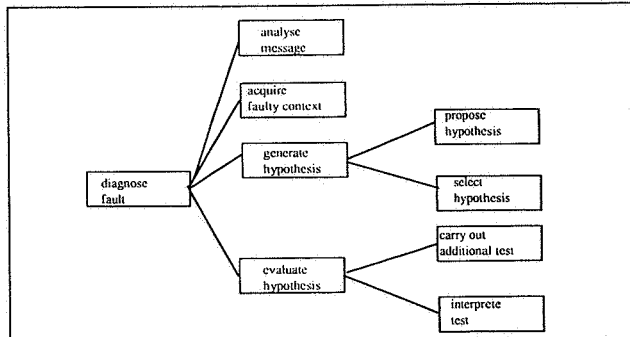


Figure 4.2. : The first stages in the diagnosis process modelled in the problem solving framework

The use of MACAO on this application facilitates the two stages scheduled in the project because the model can be increased step by step. Moreover, it helps understand better the expertise when the system is extended.

The schema of the conceptual model plays the role of a frame that guides implementation choices as well. Depending on the expert's availability, the system can be developed with more or less expertise. The least amount of knowledge required is the schema of the conceptual model. It can be used to decide whether the system can be built with a case-based or a model-based reasoning approach. It provides information on the kind of domain model used for model-based reasoning for instance.

4.2 THE SYSTEM ARCHITECTURE

We define the architecture of the diagnosis expert system as a hybrid one: it relies on rule based, case based and model based reasoning (see Figure no. 4.3).

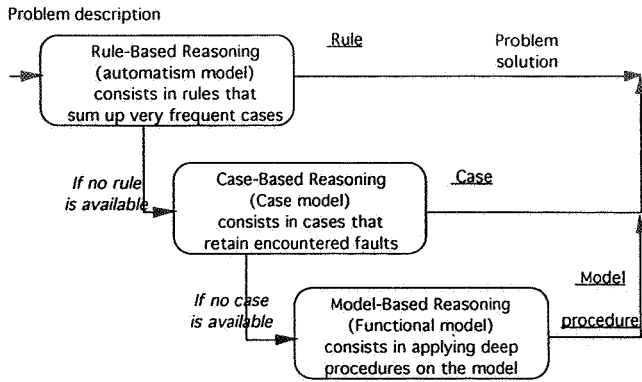


Figure 4.3 The distribution of the three model competence

Our goal is to combine AI techniques that can propose an efficient model of an expert's problem solving process. We assume suitable to combine Rule Based Reasoning, Model Based Reasoning (MBR) and Case Based Reasoning (CBR) techniques. Given an incident, the system will first rely on rules. If it doesn't succeed, CBR will be triggered to look for similar old cases. This can shortcut the model based diagnosis. Otherwise, the MBR must be triggered on the functional and structural model of the VEB. It can then assist the user or resolve the problem if its knowledge is strong enough.

4.2.1 CBR as a Component of a Hybrid Architecture

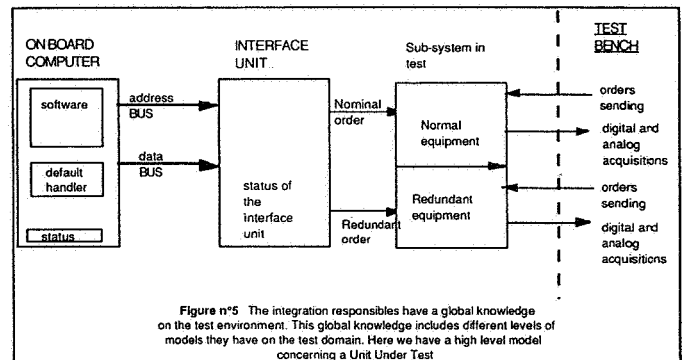
We decided to adopt case based reasoning techniques for two reasons: the expert refers to previous cases to solve new problems and many occurring incidents have already been encountered. Thus, when an incident occurs, the system will search for similar old cases to adapt the diagnosis approach to the present case. Explanation facilities concerning system reasoning will partly rely on the Case Based Reasoning architecture.

4.2.2. CBR and the Domain Model

The integration responsible has an overall view of his test environment. Thus he has a general knowledge of the system he has to test and of the test bench he uses. When an anomaly occurs, he has to find out the cause of this anomaly and to detect from which equipment it comes. In general, his diagnosis process ends when one equipment of the system in test is identified as guilty. Indeed,

it is beyond his knowledge to thoroughly detect which part of the equipment is faulty. But by experience, he can have partial knowledge on the inside of the equipment.

Thus, to ensure the modification and enrichment of our anomaly cases, we have to realize a global model of all the systems involved in the test (sub-systems in test, interface unit, test bench...) and their connections with each other. This modelling corresponds to the view the expert has on the test environment and is, in fact, constituted with different levels of models. The following drawing (figure 5) illustrates a high level model concerning a Unit Under Test. (Lower level models correspond to more details for each equipment represented in the high level model).



In addition to the description of the anomaly, we have to define all the concepts associated to the contexts in which the anomalies occur, such as symptoms, anomaly causes, and test context. Examples are: message-coming-from-the-test-bench, single-message, and sub-system-A-in-fault.

4.2.3 Case Organization in Ariane4

In Ariane4, we organize cases of anomalies in a two-level hierarchy:

At the higher level, we have a set of super cases that gather subtypes of cases. A super case is characterized by common features of all the cases that belong to it. Of course, cases under super cases can have additional features just like subclasses and classes in object oriented programming paradigms. At the lower level, we have cases with their own

specific features.

Given an incident, the system will search if there exists a super case with features that correspond with that of the incident. If this super case exists, then the system will go through the organization under this super case to localize potential similar old cases. Globally we can say that the super cases regroup information related to external symptoms that occur and the case organization under the super cases corresponds to the diagnosis approach.

4.2.4 The System Architecture and the Long Term Objective

In defining the architecture, we are particularly focusing on the possibility for the system to have its knowledge enriched and maintainable. Each time we have new knowledge to integrate in the system, we have to study its significance and impact. In effect, we can integrate this knowledge as new cases of anomalies (by using the Case Based Reasoning architecture), as new items of modelization (by using the Model Based Reasoning architecture) and as new heuristics (by translating them first at the level of the expert system conceptual model and then by coding them in the rule based architecture).

4.3 COOPERATIVE MODEL

To adapt better the system to users' cognitive and physical skills, the man-system cooperation has to be precisely defined. The cooperative process, in our approach, is organized in a three layer model:

- the nucleus is a theoretical cooperative model,
- the theoretical model is shaped by operational environment constraints,
- the users behavioral model defines the cognitive compatibility between system and users.

As a project parallel to that of corporate memory, we are building a cooperative model responding to the three above criteria. A system with such a cooperative model is capable of adapting its dialogue with the

levels of users. Thus, for instance, the information (results, explanation, and reasoning) will be presented according to the testers' level of competence.

This cooperative, for its advanced concepts, will not be implemented in the knowledge based system. But realizing it allows us to have more insight in the definition of the interface and functionalities of our expert system.

Indeed, the cooperative model allows us to take better into account constraints of the work environment, characteristics of the work organization and human factor aspects in the context of Ariane4 VEB AIT/AIV activities.

5. CONCLUSION

Introducing the expert system in the context of Ariane4 VEB AIT/AIV activities will change the work organization. Indeed, when confronted with an incident, instead of only referring to documents and to human experts, testers can use the system as a source of knowledge and consider it as a tool of their own. Besides, they know that they can improve the performance of this tool by modifying and enriching its knowledge. Hence, in such a context, the work productivity is ensured as well as the technical level of the staff is kept high owing to the introduction of such a system.

The approach we adopt in the context of Ariane4 to capitalize the corporate memory can be adapted in the context of satellite integration or in other work environment. At each environment, our approach will allow us to identify and capitalize the know-how and the experience and then to translate them in terms of advanced information systems.

With the rise of information technology, one can feel overwhelmed in finding solutions to a problem encountered in his work environment. The approach we propose allow to master the solution by having a global understanding of the problem the parameters of which range from human factors to work domain. Hence, this enables us to have more insight as to choices in technical solutions.

Moreover, we enable a smooth introduction of advance systems in work organization to have more impact on the productivity and quality of work.

Artificial Intelligence and Knowledge Based Systems for Space, May, 1991, ESTEC.

REFERENCES

Aussenac, N., Soubie, J.L., Frontin, J., Riviere, M.H. *A Mediating Representation to Assist Knowledge Acquisition with MACAO*, 3rd European Knowledge Acquisition Workshop (EKAW'89), Paris, July, 1989.

Aussenac, N., Matta, N., *Making the Method of Problem Solving Explicit with MACAO: The SISYPHUS Case-Study*, Proceedings of the 6th European Workshop on Knowledge Acquisition - EKAW 92, May, 1992, Heidelberg (Germany).

Brenot, J.M., Caloud, P., Valluy L., *On the Development of an Operational Expert System for the Telecom 2 Satellite Control Centre*, Proceedings of the ESTEC Workshop on Artificial Intelligence and Knowledge Based Systems for Space, May, 1991, ESTEC.

Feigenbaum, E., McCorduck, P., Penny, H., *The Rise of the Expert Company*, Vintage Books, New York, 1988.

Hammond, K., *Case-based Planning*, Ed. Chandrasekaran (Perspectives in AI), New York: Academic Press, 1989.

Linster, M., *Knowledge Acquisition Based on Explicit Methods of Problem Solving*, PhD Thesis, Univ. Katerslautern, D 386, Feb, 1992.

Neches, R., Fikes, R., Finin, T., Gruber, T., Patil, R., Senator T., Swartout, W., *Enabling Technology for Knowledge Sharing*, AI Magazine, Vol. 12, no. 3, Fall, 1991.

Riesbeck, C., Schanck, R., *Inside Case Based Reasoning*, LEA Assoc., Hillsdale, NJ, 1989.

Scott, A., Clayton, J., Gibson, E., *A Practical Guide to Knowledge Acquisition*, Addison Wesley, Reading, MA, 1991.

Vo, D.P., Aussenac, N., Carre, F., *Evaluation of the Impact of Operational AI on Organization*, Proceedings of the ESTEC Workshop on

NASA's Artificial Intelligence Research Project: Lessons Learned

M. D. Montemerlo
National Aeronautics and Space Administration Headquarters
Washington, D.C., USA

Presented but not submitted to Proceedings
as of December 14, 1992

N94-23900

Combining Real-time Monitoring and Knowledge-Based Analysis in MARVEL

U. M. Schwuttke, A. G. Quan, R. Angelino, and J. R. Veregge

Jet Propulsion Laboratory, California Institute of Technology,
 MS 301-270, 4800 Oak Grove Drive, Pasadena, CA 91109
 (ums@puente.jpl.nasa.gov)

ABSTRACT

Real-time artificial intelligence is gaining increasing attention for applications in which conventional software methods are unable to meet technology needs. One such application area is the monitoring and analysis of complex systems. MARVEL, a distributed monitoring and analysis tool with multiple expert systems, has been developed and successfully applied to the automation of interplanetary spacecraft operations at NASA's Jet Propulsion Laboratory. In this paper, we describe MARVEL implementation and verification approaches, the MARVEL architecture, and the specific benefits that have been realized by using MARVEL in operations.

1. INTRODUCTION

MARVEL (Multimission Automation for Real-time Verification of spacecraft Engineering Link) is an automated system for telemetry monitoring and analysis at NASA's Jet Propulsion Laboratory (JPL). MARVEL has been actively used for mission operations since 1989. It was first deployed for the Voyager spacecraft's encounter with Neptune and has remained under incremental development since that time, with new deliveries occurring every six to ten months. MARVEL combines standard automation techniques with embedded knowledge-based systems to simultaneously provide real-time monitoring of spacecraft data, real-time analysis of anomaly conditions, and a variety of productivity enhancements. The primary goal of MARVEL is to combine conventional automation and knowledge-based techniques to provide improved accuracy and efficiency and reduced need for constant availability of human expertise. A second goal is to demonstrate the benefit of incorporating knowledge-based techniques into complex real-time applications.

The traditional spacecraft operations environment at JPL has not relied heavily on automation. This is because until fairly recently, software technology was insufficient for meeting the complex needs of this

application. The traditional approach has involved large teams of highly-trained specialists and support personnel for each spacecraft subsystem and each mission. The total operations staff for the two Voyager spacecraft during peak activity periods (such as planetary encounters) has consisted of over 100 individuals. This traditional approach has been used successfully for the Voyager mission, resulting in enormous volumes of scientific data from brief fly-by encounters.

Despite the past successes, the increasing number and complexity of missions will cause this operations approach to become less feasible for two reasons. First, the workforce costs for supporting this style of operations for multiple simultaneous missions are too great to be sustained by current NASA budgets. Secondly, with the exception of Voyager, missions will be returning significantly higher volumes of engineering and science data on a more continuous basis than in the past.

MARVEL provides user-interface functions, data access, data manipulation, data display, and data archiving within an X-windows/Motif environment. The detailed expertise for anomaly analysis is implemented with embedded knowledge-based systems. In the event of anomalies, the appropriate knowledge bases provide an analysis and recommendations for corrective action. MARVEL makes it possible for an analyst to effectively handle significantly more demanding real-time situations than in the past, because it automatically performs numerous tasks that previously required human effort. As a result of MARVEL, it has become possible for individual analysts to be responsible for several spacecraft subsystems during periods of low and moderate spacecraft activity. This is because MARVEL reduces both the level of training and the cognitive load that are required to perform routine operations.

MARVEL has demonstrated that automation enhances mission operations. Individual spacecraft analysts are no longer burdened with routine monitoring, in-

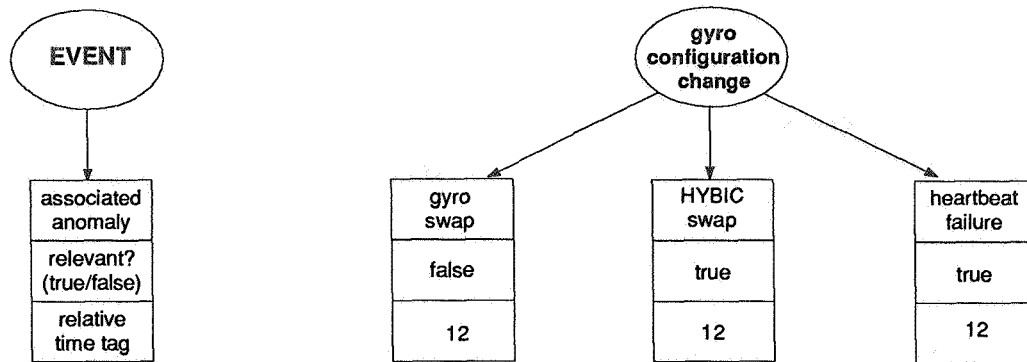


Figure 1. The event structure. The figure on the left depicts the general form of the structure; the figure on the right shows a specific instance of an event associated with three different anomalies.

sistent data rather than on a single piece of data in isolation.

2.3. Temporal Reasoning with Minimal Impact on Real-Time Processing

Real-time systems often need to reason about past events and about the order in which they occurred. The MARVEL expert systems respond to events (symptoms) indicated in the spacecraft telemetry by attempting to identify and diagnose specific subsystem anomalies that caused an event. In order to do this, the expert system may need to know about other spacecraft events that have occurred in the past and the sequence of their occurrence. This involves temporal reasoning, which is implemented in MARVEL using dynamically updated structures, as shown in Figure 1.

The structures contain the name of the event, the name of an anomaly that may have caused the event, a Boolean flag indicating whether the event has occurred and is currently relevant, and an integer specifying the sequence in which the event occurred relative to other events. The anomaly identifier is necessary since a particular event may have bearing on the diagnosis of more than one anomaly (that may or may not have occurred). Thus, a single event may point to multiple structures that are each associated with a different anomaly. The Boolean flag is set when the event associated with the structure is detected from telemetry. When this occurs, the relative time of the event is recorded in the structure. The validity of a Boolean flag expires after its corresponding anomaly is resolved, causing the flag to be reset so that it cannot contribute to the detection or diagnosis of the same anomaly unless the associated event occurs again.

These structures are intended to have minimal impact on performance. Once an event is detected, a structure

is created for each anomaly whose diagnosis may depend on that event. Thus the multiple pieces of evidence that confirm the occurrence of an event need only be evaluated once, regardless of how many anomalies may be related to that event. Also, event structures are not retained indefinitely. There is a time limit beyond which an event structure is considered no longer useful for identifying and diagnosing new anomalies. After this time limit has expired, a structure's Boolean flag is reset to false regardless of whether its associated anomaly has been diagnosed. This minimizes the number of event structures that are active or relevant at any one time, which in turn reduces the number of event structure comparisons that must be performed during a rule evaluation cycle.

2.4. Multiple Knowledge Bases For Improved Focus of Attention

When significant events occur, real-time knowledge-based systems must focus their attention and resources on relevant parts of the search space in order to achieve adequate performance. Many expert system environments do not have an efficient method for doing this. One standard way to enable focus of attention is to apply different subsets of the domain rules within different contexts. MARVEL accomplishes this with separate knowledge bases for each spacecraft subsystem, and with rule contexts (mini-experts) within the individual knowledge bases.

A top-level data management process identifies incoming telemetry and determines which subsystem monitoring module to invoke for anomaly detection. When an anomaly is found, the subsystem monitor then invokes its corresponding expert system to perform the necessary analysis. This logical partitioning of input data among reasoning modules enables more rapid traversal of the search space and helps to ensure

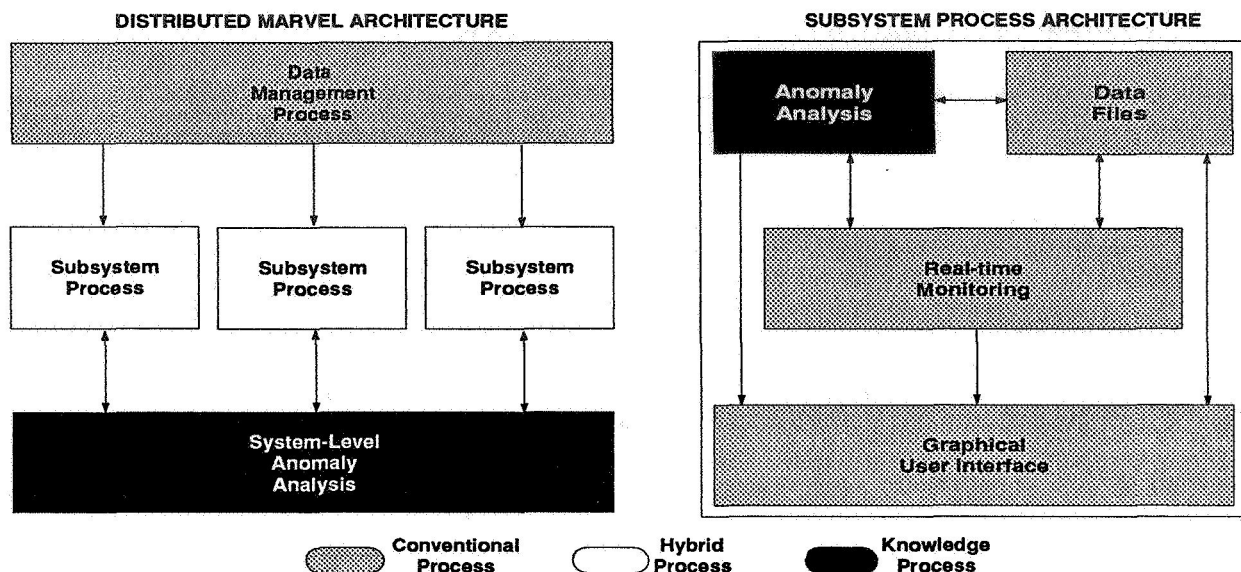


Figure 2. The MARVEL architecture, shown at the left, consists of knowledge processes, conventional processes, and hybrid processes. It can be configured to run on one to four workstations, depending on operations needs. The subsystem architecture shown at the right provides a more detailed look at the structure of MARVEL's hybrid subsystem processes.

that conclusions and responses that are not relevant to the current analysis are not pursued. This approach has also contributed to maintainability of the knowledge bases, in that several smaller knowledge bases are easier to maintain than a single large one.

3. THE DISTRIBUTED ARCHITECTURE

There are many reasons for distributed problem solving [Bond 1988]. For example, distributed systems are often characterized by greater computational speed because of concurrent operation. Also, a distributed system can be significantly more cost-effective, since it can include a number of simple modules of low unit cost. Further, distributed systems may offer a more natural way to represent certain classes of problems which contain inherently partitionable sub-processes. Each of these reasons is considered important in the mission operations environment, and as a result, a distributed MARVEL environment has been implemented.

3.1 Implementation

The distributed MARVEL architecture shown in Figure 2 is based on a central message routing scheme. The various software modules are allocated among a configuration of UNIX workstations. The data management module receives telemetry data from JPL's

ground system. Each of the three subsystem monitors provide functions such as validation of telemetry, detection of anomalies, diagnosis of causes, and recommendation of corrective actions. The latter two functions are provided through intelligent reasoning modules that are embedded within each of the individual subsystem monitors. The remaining modules include the display processes for each of the three subsystem monitors, and the system-level reasoning module for diagnosing anomalies that manifest themselves in multiple subsystems (and therefore cannot be completely analyzed by any one subsystem alone).

The interconnectivity of the distributed system is provided by a TCP/IP central router program and a set of messaging routines that are linked into the subsystem processes. All MARVEL processes are connected to the central router by UNIX sockets.

3.2 Performance

For realistic systems with non-negligible communication overhead, the critical measure curve is related to the speedup $S(N)$ [Fox 1988] defined as

$$S(N) = T_{seq} / T_{conc}(N).$$

In this equation, N denotes the number of processors, and T_{seq} and $T_{conc}(N)$ refer to the execution times of the sequential program and the distributed program on N processes, respectively. Distributed systems

with a speed-up $S(N) = 0.8N$ are considered to be very efficient [Fox 1988]; the minimum desired speedup for a distributed MARVEL system was $0.6N$.

The basic measurement of performance for the distributed MARVEL is $S(N)$. However, it has not been possible to measure a unique value $S(N)$ because of the heterogeneous nature of the MARVEL modules. This heterogeneity arises because the processing load of the four basic components (the data management module and the three subsystem modules) are not identical. Our alternative to this measurement is the lowest speedup of the individual subsystems. With a four-processor implementation, a speedup of 3.6, or $0.9N$ was observed. This result indicates that MARVEL is a highly efficient distributed system. Two factors contribute to the success of these results. The first of these is the modularity inherent in the application (as is common in many other complex applications). The second factor is a distributed design that effectively minimizes the need for interprocess communication.

4. DISCUSSION

The development of MARVEL has shown the value of a rapid development approach that emphasizes top-down design and bottom-up implementation. The implementation has been modular and incremental, with frequent deliveries (every 5 to 10 months) of new or enhanced capabilities. The result has been an automated tool that began as a simple software module for automating straightforward tasks and that has evolved over a period of five years into a sophisticated system for automated monitoring and analysis. The initial modular design enabled MARVEL to be developed incrementally, with each subsequent delivery providing greater breadth to the application. This approach has been instrumental to the success of the effort, because it was compatible with available budgets and encouraged user and sponsor confidence with frequent demonstration of results. In addition, the approach has influenced the validation and use of MARVEL as described below.

4.1 Verification of Expert Systems

MARVEL verification has been ad hoc, largely because of a lack of formal procedures. Two methods have been used: carefully engineered test cases and on-line verification (involving parallel operations with human analysts). Most problems were detected with the use of test-cases, but some were not detected until

the software was used in an on-line mode. Newly delivered modules were subject to an on-line verification period, typically on the order of one month. On-line verification allowed continual comparison with the results of manual approaches so that reasonable confidence in the automated system could be obtained.

The primary advantage of this approach has been its minimal impact on development costs. However, there have also been several disadvantages. On a few occasions, minor bugs in MARVEL went undetected until the end of the parallel-operations phase. This temporarily undermined user-confidence, particularly with users who were not enthusiastic about automation. A second disadvantage is that without formal verification procedures there have been occasional questions about whether MARVEL should be formally accepted as "official" ground software for mission operations. The current lack of solid answers in this area would prevent the use of MARVEL's AI modules for certain tasks that are considered mission critical, but has not prevented the use of these modules in an advisory mode.

4.2 Use and Benefit of MARVEL

MARVEL has been in active, daily use since it was first deployed in 1989. The current version performs real-time monitoring for three spacecraft subsystems. These functions previously required the presence of human analysts for a minimum of eight hours per subsystem per day. During planetary encounters, human presence was required on a twenty-four hour basis. MARVEL also performs non-real-time functions that were previously unautomated. These functions save from 30 minutes per week (for clock drift analysis) to 2 hours per day (for daily report generation). During MARVEL's on-line tenure, it has detected all the anomalies that occurred within its domain. During parallel operations, most anomalies were first detected by MARVEL. On two occasions MARVEL detected anomalies that operations personnel believe may have been overlooked by them because the quantities of data transmitted at those times were larger than could reasonably be handled without automated assistance.

Initial emphasis on productivity enhancement resulted in an early version of MARVEL that (according to the responsible operations supervisor) would have made real-time CCS subsystem workforce reductions of 60% (3 out of 5 analysts) possible during the Nep-

tune encounter, had MARVEL been approved for stand-alone rather than parallel operations. Subsequent to the Neptune encounter, significant workforce reductions have been implemented for all spacecraft subsystems because of post-encounter budget cuts. MARVEL played a substantial role in simplifying the transition to reduced workforce for the subsystems for which it was available.

The initial emphasis on productivity enhancement temporarily curtailed the development of MARVEL's expert systems, because it was perceived that diagnostic systems did not improve efficiency of operations. This perception stemmed from the observation that anomaly analysis was only required in the presence of spacecraft anomalies, which did not occur with sufficient frequency to warrant an automated approach, particularly since human confirmation of the expert system analysis would still be required.

However, the post-encounter workforce reductions caused renewed interest in expert systems. However, the goal is no longer workforce reduction, but the preservation of mission expertise. The current analysts are new to the mission and, for the most part, do not have the experience of the previous staff. The new personnel will have fewer opportunities to gain such experience: although the Voyager interstellar mission is scheduled to continue until approximately 2018, spacecraft activity is at a low level. This means that there are far fewer opportunities for learning about the spacecraft and its operation. There is concern that analysts with the experience to handle future anomalies will be less readily available, or that they will have retired. As a result, MARVEL's expert systems are being expanded to provide information that is based on the expertise of former analysts.

6. SUMMARY

This paper has presented methods for combining conventional software with AI techniques for use in real-time problem solving systems. The methods described have been presented in the context of the MARVEL system which has provided a continuous and evolving demonstration of the success of the approach since Voyager's Neptune encounter in August 1989. These techniques have been implemented in a distributed environment that will accommodate the real-time demands of NASA's more recently launched interplanetary missions.

7. ACKNOWLEDGMENT

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology under a contract with the National Aeronautics and Space Administration. The authors wish to acknowledge support from JPL's Voyager Project, and Director's Discretionary Fund.

8. REFERENCES

- Bahr, E. and Barachini, F. 1990. Parallel PAMELA on PRE. In *Parallel Processing of Engineering Applications*, ed. R. A. Adey, 209-219. New York: Springer-Verlag.
- Barachini, F. and Theuretzbacher, N. 1988. The Challenge of Real-time Process Control for Production Systems. *Proceedings of AAAI-88*, 705-709.
- Bond, A. H., and Gasser, L. 1988. *Readings in Distributed Artificial Intelligence*. San Mateo: Morgan Kaufmann.
- Chandrasekaran, B. and Punch, W. 1987. Data Validation During Diagnosis: A Step Beyond Traditional Sensor Validation. In *Proceedings of AAAI-87*, 778-782.
- Fox, G., et al. 1988. *Solving Problems on Concurrent Processors*, Volume I. New Jersey: Prentice Hall.
- Hayes-Roth, B. 1990. Architectural Foundations for Real-time Performance. *Artificial Intelligence Journal*, 26: 251-232.
- Horvitz, E. J.; Cooper, G. F.; and Heckerman, D. E. 1989. Reflection and Action Under Scarce Resources: Theoretical Principles and Empirical Study. *Proceedings of the Eleventh International Joint Conference on Artificial Intelligence*, 1121-1127.
- Schwuttke, U. M. and Gasser, L. 1992. Real-time Metareasoning with Dynamic Tradeoff Evaluation. *Proceedings of AAAI-92*.
- Schwuttke, U. M.; Quan, A. G.; Angelino, R; et al 1992b. MARVEL: A Distributed Real-time Monitoring and Analysis Application. In *Innovative Applications of Artificial Intelligence*, Volume 4, ed. C. Scott and P. Klahr, Boston: MIT Press. 1992.

N94-23901

THE APPLICATION OF AUTOMATED OPERATIONS AT THE INSTITUTIONAL PROCESSING CENTER

THOMAS H. BARR

OA0 CORPORATION
ALTADENA, CA, USA

SUMMARY

The JPL Institutional and Mission Computing Division (37); Communications, Computing and Network Services Section (372); with its mission contractor, OAO Corporation, have for some time been applying automation to the operation of JPL's Information Processing Center (IPC). Automation does not come in one easy to use package.

Automation for a data processing center is made up of many different software and hardware products supported by trained personnel. The IPC automation effort formally began with console automation, and has since spiraled out to include production scheduling, data entry, report distribution, online reporting, failure reporting and resolution, documentation, library storage, and operator and user education, while requiring the interaction of multi-vendor and locally developed software.

To begin the process, automation goals are determined. Then a team, including operations personnel, is formed to research and evaluate available options. By acquiring knowledge of current products and those in development, taking an active role in industry organizations, and learning of other data center's experiences, a forecast can be developed as to what direction technology is moving. With IPC management's approval, an implementation plan is developed and resources identified to test or implement new systems. As an example, IPC's new automated data entry system was researched by Data Entry, Production Control, and Advance Planning personnel. A proposal was then submitted to management for review. A determination to implement the new system was made and elements/personnel involved with the initial planning performed the implementation. The final steps of the implementation were educating data entry personnel in the areas effected and procedural changes necessary to the successful operation of the new system.

There is danger in today's market place of purchasing automation products that may become

antiquated within months, or purchasing products that may not be cost effective, or losing sight as to what future requirements our data center may be called upon to support. The IPC has chosen the commercially available and vendor supported software product, OPS/MVS by Legent, to serve as the corner stone to JPL's automation effort. Other vendors, such as, Computer Associates, Diversified Software, Unitech, and TSI also have provided software products that compliment JPL's automation effort. Hardware obtained from AGI, AST, HP, Symbols/MSI, and others have been used to support automated systems. Some systems have been developed in-house by IPC personnel to tailor automation to our specific needs.

Demands upon today's data centers are increasing: there is more processing to be done; the processing is more complex and demanding; user satisfaction and quality requirements must be achieved or exceeded; and there is an on-going demand for cost containment. The application of automated operations is the IPC's approach to achieving these demands.

A sample description of the IPC's automation activities and benefits realized is as follows:

- Automated Production Scheduling
An automated tool obtained from Computer Associates, CA-7, has replaced the manual submission of production job scheduling. This tool and associated processes/procedures ensures that "the right job is run at the right time". It also helps balance work loads. As a result, reruns have been reduced, schedules are met and processing time/resources are optimized. The IPC has accommodated a 60% growth in production scheduling requirements with no impact on resources, which was made possible by the application of automated scheduling.

Automated Operations at the IPC is essential in maintaining cost effectiveness, ensuring user satisfaction and quality and increasing production through-put.

Key words: Computer Operations, Automated Operations, Information Processing Center, mainframe processing, console automation.

1. INTRODUCTION

The Jet Propulsion Laboratory, Institutional and Mission Computing Division; Communications, Computing and Network Services Section; with its mission contractor, OAO Corporation, have for some time been applying automation to the operation of JPL's Information Processing Center (IPC). The IPC supports all JPL efforts requiring computing or communications. The IPC provides mainframe computing and user support for JPL's flight project efforts, as well as administrative computing requirements. All these support functions are engaged in some degree of automation. Automation does not come in one easy to use package. Automation for a data processing center is made up of many different software and hardware products supported by trained personnel.

The IPC automation effort formally began with master console automation, and has since spiraled out to include production scheduling, data entry, output delivery, report balancing, and documentation. Beside the interaction of multi-vendor systems, in-house systems have been developed for failure tracking and library storage.

1.1 Automation Implementation

The automation process began with operations management making the determination to move forward with the process. Next, a team, including operations personnel, was formed to research and evaluate available options. By acquiring knowledge of current products and those in development, taking an active role in industry organizations, and learning of other data center's experiences, a forecast can be developed as to what direction industry and technology are moving. With management's approval an implementation plan is developed and resources identified to test or implement new systems. As an example, a PC based, networked attached, data entry system was researched and evaluated by Data Entry, Production Control, and

Advance Planning personnel. A proposal was then submitted to management for review and approval. A determination to implement a new system was made and the same personnel involved with the initial research and planning took an active role in the implementation and training of personnel. That system is called KeyMaster, by TSI.

There is danger is today's market place of purchasing automation products that may become antiquated within months, or purchasing products that may not be cost effective, or losing sight as to what future requirements the data center may be called upon to support. It is very important to have an automation team to research and evaluate. And it is equally important to have management review and question the automation team's proposals.

1.2 Console Automation

The IPC chose the commercially available and vendor supported product, OPS/MVS by Legent (formally Goal Systems) to serve as the console automation tool.

Console automation was the first step taken by the IPC to automate some of the functions carried out by Operations. Several thousand messages and repetitive reply situations made the master console an obvious beginning. The automation package was installed, operator training was given, and results were very slow. We learned that the operators were not too happy with the idea of automating what they had been doing manually. Performing a quiesce in 8 minutes or an IPL in 4 minutes did not impress the operations staff. And they were right, it was not impressive. We had failed to show the operations staff that automation was not replacing them. Rather it is changing their job descriptions. The computer operations profession is under going a change. Today some computer operators do not operate equipment, but monitor software that operates equipment, operators do not run to mount a tape but sit at a desk analyzing an operations anomaly. Soon operators will not load paper into large printers but will answer user requests for specific operations support. Operations is changing and the staff must change too. Operators must develop into operation analysts, help desk personnel, production control analysts, and operations managers. The process may be slow and aggravating at times, but it should not be stopped.

1.3 Automated Scheduling

Two years ago the IPC submitted scheduled production jobs manually. An operator manually started each job and ran most of the schedule single thread. Management realized that this situation was not efficient, nor farsighted. An automated scheduling package, CA-7 by Computer Associates, was chosen, purchased, and implemented. Operations staff received training, one operator became a scheduler, a support person also became a scheduler. Two members of the operations staff had made the change to using automated software to carry out what had been a manual function. After two years, our production schedule has grown by 60% and is continuing to grow, but staffing has remained at the same level and processing is being completed within the same processing window.

1.4 Balancing

If we were to automate scheduling then why not balancing. These two functions interface. IPC supported the purchase and use of U/ACR by Unitech. This automation package is largely used by the applications groups, with very little manual interfacing with operations. U/ACR is an automated balancing tool that interfaces with CA-7. If a job is out of balance then the scheduling software is made aware and the job is stopped, as well as, any succeeding jobs that use output from the out-of-balance job. A failure is recorded and at that point a problem analyst takes over.

1.5 Documentation

The next area of concern was documentation. Everyone knows, documentation is the last thing done, and very often never done. The problem was how to continually update documentation that must be made available to operations staff, applications staff, and management. The IPC chose DOCU/TEXT, by Diversified Software, to help solve this problem. We felt by using an on-line documentation system that updates could be made faster and more easily, and that up-to-date documentation would be available to everyone who needed it. The documentation is available to everyone who needs it. But it was necessary to assign operations personnel to monitor the system for incomplete documentation. Some of our documentation problem has been solved, but not all. We are still researching and discussing options to improve upon documentation maintenance.

1.6 Output Delivery

Express Delivery has been in use at the IPC for years, but is now being teamed up with SAR (Sysout Archive and Retrieval). Express Delivery is a software package that automatically addresses reports to the proper person or place. Its primary usage is with hard copy output. SAR, is a on-line mainframe storage tool for retrieval and analysis of output. By interfacing Express Delivery and SAR we hope to eliminate hard copy generation, provide immediate access to output, provide electronic archiving of sensitive and legal information, and reduce overall costs of report generation and delivery. This is IPC's effort to bring about on-line report viewing.

1.7 BLITS

BLITS, is Barcoded Library Inventory and Tracking System. It is a PC based system for JPL's Archive Tape Library, written in-house by one of our planning analysts. The IPC's Archive Library has over 120,000 magnetic tapes. The library services 25 different JPL tape libraries. The Archive Library is maintained by one person, using BLITS. BLITS uses a Portable Data Terminal (PDT), programmed in UBASIC, to read bar code labels attached to each tape. The information is downloaded to the PC in ASCII text format. The ASCII is then converted to a data base file (.DBF), which is used by several dbFast Windows programs. With these programs, inventory is tracked and reports are generated.

1.8 Failure Reporting

The semi-automated failure system was developed by Production Control and Systems personnel, and coordinated by Advance Planning. The failure system uses the automated scheduling tool, CA-7, to submit a job that identifies a failure and captures system data concerning the failure. The data is loaded to a data base, then accessed by an analyst, who records actions taken to resolve the failure. The failure information is then printed at a distributed printer, and after review, fax'ed to Configuration Management for distribution to the responsible areas. Resolution data is then entered into the failure data base. Management and statistical reports are generated, by programs written in DYL280. We call this system semi-automated because we have not yet finished implementing the on-line viewing portion of the system. When

finished the system will not use paper, but instead failure data, statistical report data, and historical data will be available on-line.

2.0 BENEFITS

Each of these tools is part of an integrated automation effort. Our goal is to not eliminate personnel but to allow increased workloads to be carried out at current labor levels. Personnel receive training for each automation tool as it is added. Soon our personnel will no longer be just tape hangers, but also help desk persons, problem analysts, automation analysts, and trainers.

The benefits we hope to achieve for automated operations are:

- less operational costs
- fewer system outages
- improved problem recognition
- faster response time
- elimination of labor intensive tasks
- improved accuracy
- reliable handling of time-dependent tasks
- reduced system outages

2.1 Current and Future Development

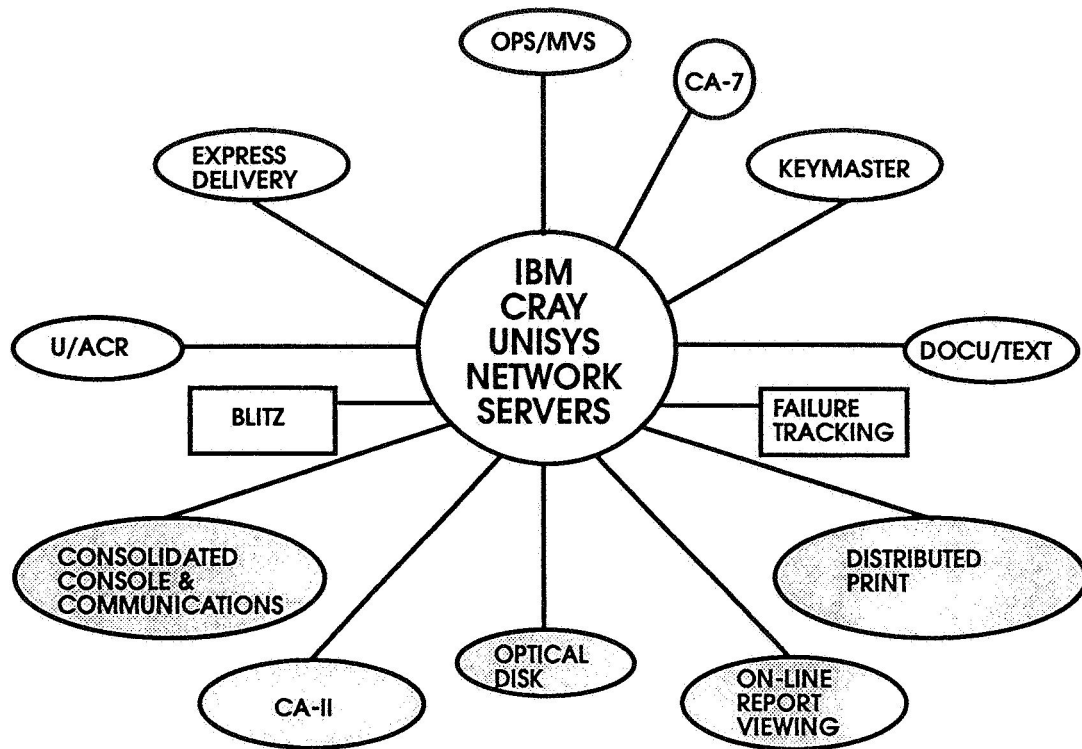
Automation is an on-going process. Currently we are developing user training classes to support the implementation and use of on-line report viewing. Development of a distributed print plan is underway. Distributed print will allow users to print output at their locations, thus reducing distribution labor and transportation time and costs. A PC based system called ELIXIR, which is used to tailor and create forms in-house, is now in use. Magneto-optical disk storage is being researched and evaluated for possible use in supporting on-line report viewing and archive storage. CA-11, a restart and rerun tool is currently being implemented. Two automated tape libraries (Silos) have been installed and are in use.

There is a plan in development to consolidate voice communications and console operations for an IBM 3090/200, Cray YMP, 2 Unisys 1100/90's, and network and client/server monitoring. Some of the development input was obtained by going outside operations to enlist the help of skilled personnel in other areas within JPL, to help us better communicate our plans and proposals. This plan is currently being reviewed by IPC management.

3.0 CONCLUSION

Demands upon today's data centers are increasing: there is more processing to be done; the processing is more complex and demanding; user satisfaction and quality requirements must be achieved or exceeded; and there is an on-going demand for cost containment. The application of automated operations is the IPC's approach to achieving these demands. Automated operations at the IPC is essential in maintaining cost effectiveness, ensuring user satisfaction and quality, and increasing production through-put.

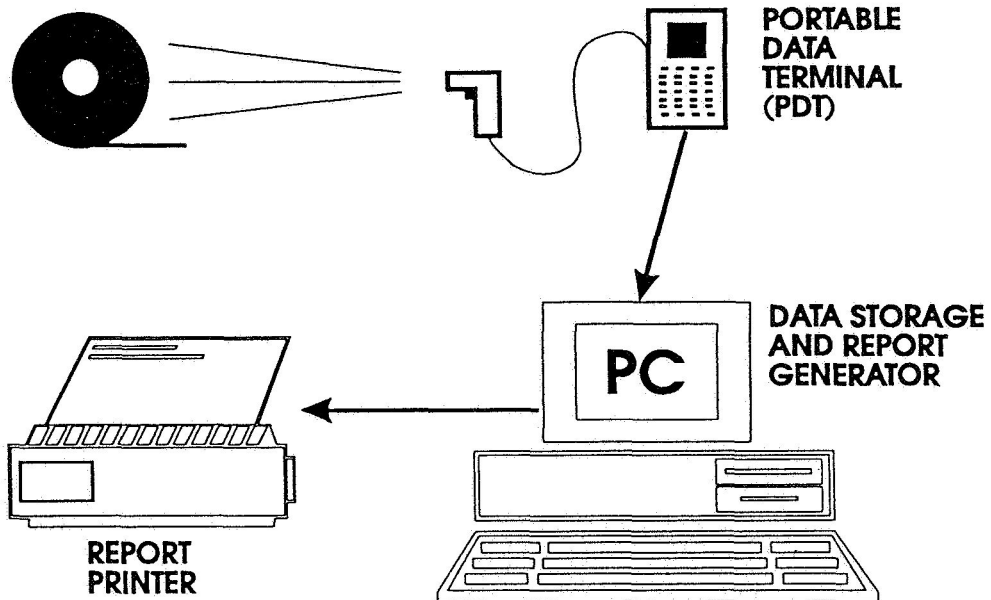
VENDOR, IN-HOUSE, AND SYSTEMS UNDER DEVELOPMENT



BLITS

BARCODED LIBRARY INVENTORY AND TRACKING SYSTEM

120,000 TAPE ARCHIVE LIBRARY



N94-23902

**Making Tomorrow's Mistakes Today:
Evolutionary Prototyping for Risk Reduction and Shorter Development Time**

by Gary Friedman, Ursula Schwuttke, Scott Burliegh, Sanguan Chow, Randy Parlier, Lorraine Lee, Henry Castro, and Jim Gersbach

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109

Introduction

In the early days of JPL's solar system exploration, each spacecraft mission required its own dedicated data system with all software applications written in the mainframe's native assembly language. Although these early telemetry processing systems were a triumph of engineering in their day, since that time the computer industry has advanced to the point where it is now advantageous to replace these systems with more modern technology.

The Space Flight Operations Center (SFOC) Prototype group was established in 1985 as a workstation and software laboratory. The charter of the lab was to determine if it was possible to construct a multi-mission telemetry processing system using commercial, off-the-shelf computers that communicated via networks. The staff of the lab mirrored that of a typical skunk works operation -- a small, multi-disciplinary team with a great deal of autonomy that could get complex tasks done quickly.

Although today many computer manufacturers are committed to converging on software standards and are scurrying to provide a common interface, seven years ago this wasn't the case. UNIX System III was ruled by AT&T, Sun was not a major player in the then-new workstation market, and TCP/IP was predicted to be dead by the Department of Defense in less than two years.

In an effort to determine which approaches would be useful, the prototype group experimented with all types of operating systems, inter-process communication mechanisms, network protocols, packet size parameters. Out of that pioneering work came the confidence that a multi-mission telemetry processing

system could be built using high-level languages running in a heterogeneous, networked workstation environment. Experience revealed that the operating systems on all nodes should be similar (i.e., all VMS or all PC-DOS or all UNIX), and that a unique Data Transport Subsystem tool needed to be built to address the incompatibilities of network standards, byte ordering, and socket buffering.

The advantages of building a telemetry processing system based on emerging industry standards were numerous: by employing these standards, we would no longer be locked into a single vendor. When new technology came to market which offered ten times the performance at one eighth the cost, it would be possible to attach the new machine to the network, re-compile the application code, and run. In addition, we would no longer be plagued with lack of manufacturer support when we encountered obscure bugs. And maybe, hopefully, the eternal elusive goal of software portability across different vendors' platforms would finally be available.

The Prototype Group (later renamed the Flight Projects Office Information Systems Testbed, or FIST) performed an evolutionary prototype of this new architecture, uncovered potential problems, and learned a great deal about multi-vendor interoperability. All of this heavily influenced the eventual implementation of the world's first distributed multi-mission telemetry processing system.

Looking back on that experience, one marvels at the risky decisions that were made to standardize on UNIX, TCP/IP and C. These decisions have since been validated by UNIX vendors who are all clamoring to move to "Open Systems" in order to strengthen their market share.

Since that time, FIST has been responsible for keeping JPL's flight projects abreast of emerging technologies, such as user interface and data visualization tools, networking, artificial intelligence, distributed systems, databases, and other promising software technologies in order to determine how these might benefit both the mission operations and scientific communities. We showed Voyager scientists real-time graphical data during the Neptune encounter. We've demonstrated the feasibility of graphical planning tools and shown mission operations teams how today's tools can be applied to make their day-to-day activities more manageable. But most importantly, FIST has been a resource that can help JPL remain cutting edge with minimal cost and risk.

Some highlights of our prototyping efforts are described in more detail later in this paper

The Advantages of Evolutionary Prototyping

Evolutionary prototyping refers to high-quality programs used to validate or uncover requirements, and to validate a possible design. Unlike a throw-away prototype, which is useful when many of the aspects of a design are untried, evolutionary prototypes are robust in design and are built upon a foundation of that which is well-understood. Evolutionary prototypes are designed to be built upon, incorporating foresight and software hooks to allow for expansion and eventual delivery.

Why do evolutionary prototyping? Three primary reasons: risk reduction, shortened development cycle, and accelerated technology transfer.

The idea of pre-development evolutionary prototyping has already proven its value to many JPL projects. It is a fast and cost-effective method of proving out new concepts and accelerating their simultaneous integration into operational environments and next-generation products.

Prototyping can hasten the transfer of new technology into large systems. The short-loop feedback cycle from operational settings and user communities results in real user requirements being dynamically defined. The result is a useful product in a shorter time and

without the inherent risk of untried technologies. Evolutionary prototypes require less documentation, comply with limited formal specifications, and involve informal configuration management and fewer formal verification tests.

In addition, this type of prototyping concludes in the worst case with a small-scale, limited distribution product for operational environments, and in the best case leads to full-scale development of multi-user and multi-mission systems.

Highlights of FIST Prototypes and Pilots

VNESSA

To support Voyager II's historic encounter with Neptune, the FIST lab prototyped a new science visualization system called VNESSA (Voyager Neptune Encounter Science Support Activity). This pilot project encompassed both the user interface as well as the telemetry processing engine needed to drive it, and was the first time any of FIST's prototypes presented a system view rather than individual pieces of the telemetry processing puzzle.

VNESSA represented several "firsts" for Voyager. This was the first time that fields and particles investigators had access to their data electronically in near-realtime. Recently acquired near-realtime data were also stored in a database for the first time, and could be recalled later. It was also the first time that this data could be displayed in multiple, animated color graphical "windows" in a computer screen under the control of the science user. For the first time, VNESSA allowed these investigators to provide graphical illustrations of their discoveries to the news media almost immediately after receiving the near-realtime data. There was even a speech synthesizer connected to the operator's console which allowed operators to audibly monitor critical events even when they were elsewhere in the support area.

The science teams used VNESSA more heavily during the encounter than was expected, and the system's usefulness exceeded the expectations of most participants. The VNESSA-developed data displays were unexpectedly popular for correlating results from two or more instruments, verifying the science

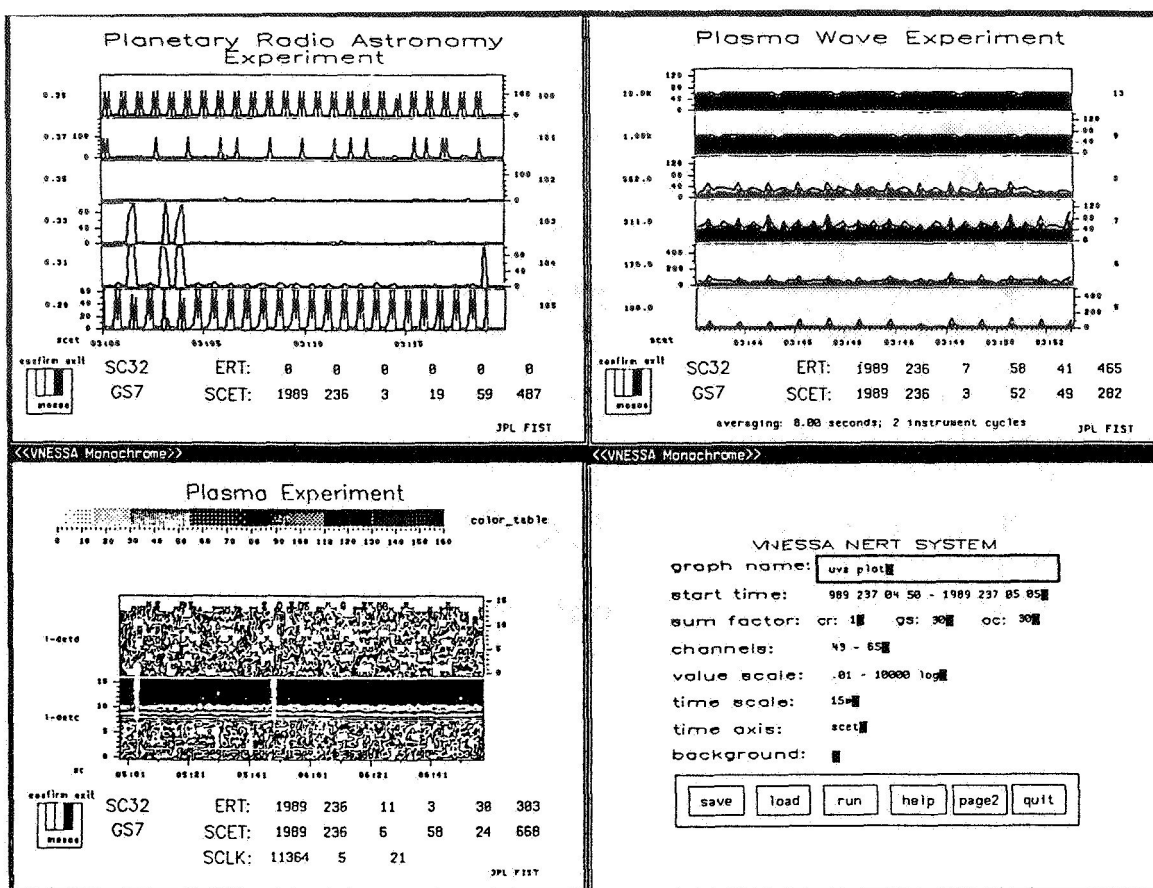


Figure 1. VNESSA provided Voyager Principal Investigators with access to Neptune encounter science data in near real time and provided enhanced near real time science data processing capabilities. Three VNESSA displays for the Neptune encounter show plots for Planetary Radio Astronomy (PRA) and Plasma experiments.

teams' own analyses, and supporting members of teams whose own equipment could not accommodate any more users. We also accidentally discovered that putting all the Principal Investigators and their near-real-time data displays into one room can catalyze scientific discovery: one team's discovery of a milestone event can confirm and enhance the conclusions of the other teams.

Lessons Learned from VNESSA

This prototype taught us a lot about building real-time high-volume telemetry data servers on top of commercial database products which were designed for more conventional use. Considerable energy was devoted to developing work-arounds to these database problems, most of which were caused by unusual Instrument Data Frame lengths and volumes. Other lessons learned fell into the hindsight category: although we started 18 months early and had regular dialog with the science teams, FIST archives proclaim that we should have started sooner, should have had more interaction among participants, should have placed VNESSA workstation displays in the Voyager General Science

Data Team's work area.

Overall, VNESSA reinforced the notion that, using commercial tools and de facto industry standards, a small, skilled, and stable development team can perform impressive feats and rapidly usher in new technology much faster than normally would have occurred.

SANTA

To support the Galileo spacecraft's first two major events, FIST also performed evolutionary prototyping to create a UNIX-based telemetry processing system which worked in parallel with the original Galileo data stream. The name SANTA (Science Analysis Near-Term Activity) was chosen since both activities - the initial 4-day instrument checkout and the "Earth-1 encounter" (the first time Galileo passed near the Earth on its gravity-assisted trajectory to Jupiter) - occurred near Christmas time in their respective years.

The first Science Analysis Near-Term Activity

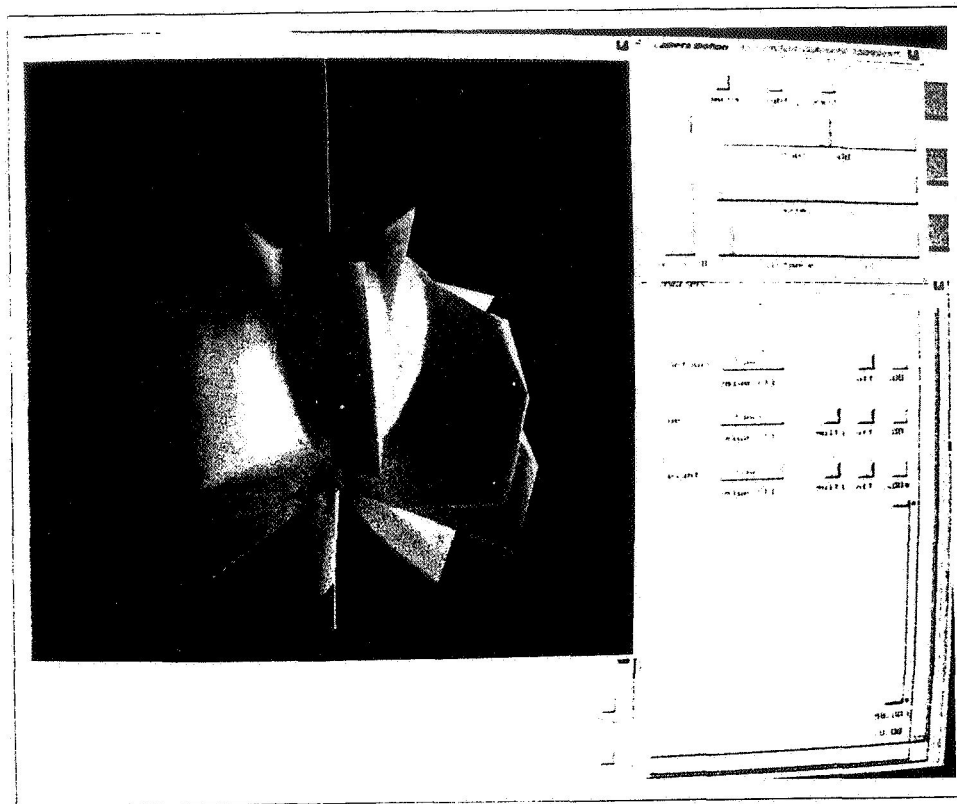


Fig. 2 *SANTA II* was prototyped to support Galileo's Earth-I encounter. It delivered low-rate science in near real time to geographically dispersed Principal Investigators in a secure fashion over the Internet. In addition, many new visualization schemes were tried. Above is a prototype "3-D" phase-space plot of plasma particles. Each picture represents a particle type sampled using one revolution of the spacecraft's spun bus.

(SANTA) was a demonstration of the quick-look science support component planned for the Space Flight Operations Center. To accomplish this goal, the VNESSA Instrument Data Frame server was adapted to process Galileo data. It accurately delivered low-rate science data to Galileo's scientists in real time (generally about 6 seconds after receipt at Earth) for four days; during this time it dropped only two minutes of recoverable data.

The Galileo scientists were reportedly pleased with the system and very interested in using this kind of real-time science data access extensively in the future, both for the upcoming Galileo encounters and for other space flight missions.

SANTA II (December, 1990) was built for the Galileo Earth-I encounter, and strove to do for Galileo what VNESSA had done for Voyager: show the project the kinds of things possible when using the SFOC architecture. The project successfully brought together talent from three different JPL sections, half a million dollars' worth of loaned computer equipment, and a pre-alpha version of a newly-redesigned SFOC.

SANTA II had a wide range of aspirations, among them prototyping of new and "equivalent" science displays, demonstrating new SFOC subsystems, trying out new computer technology, and the secure distribution of data via the Internet. Like VNESSA, we had near-real-time science displays whose parameters were user-definable.

The area of largest impact was the controversy over security procedures. Users wanted seamless, remote, automated access to our telemetry data server, while management, concerned about external system break-ins and viruses, insisted on some mechanism to ensure access only to valid users. A brain-dead UNIX box (with all the net.daemons turned off and only one active network port) was used in conjunction with challenge-response authentication tokens, which required the users' presence and precluded any automation of the data gathering process.

Lessons Learned from SANTA I and II

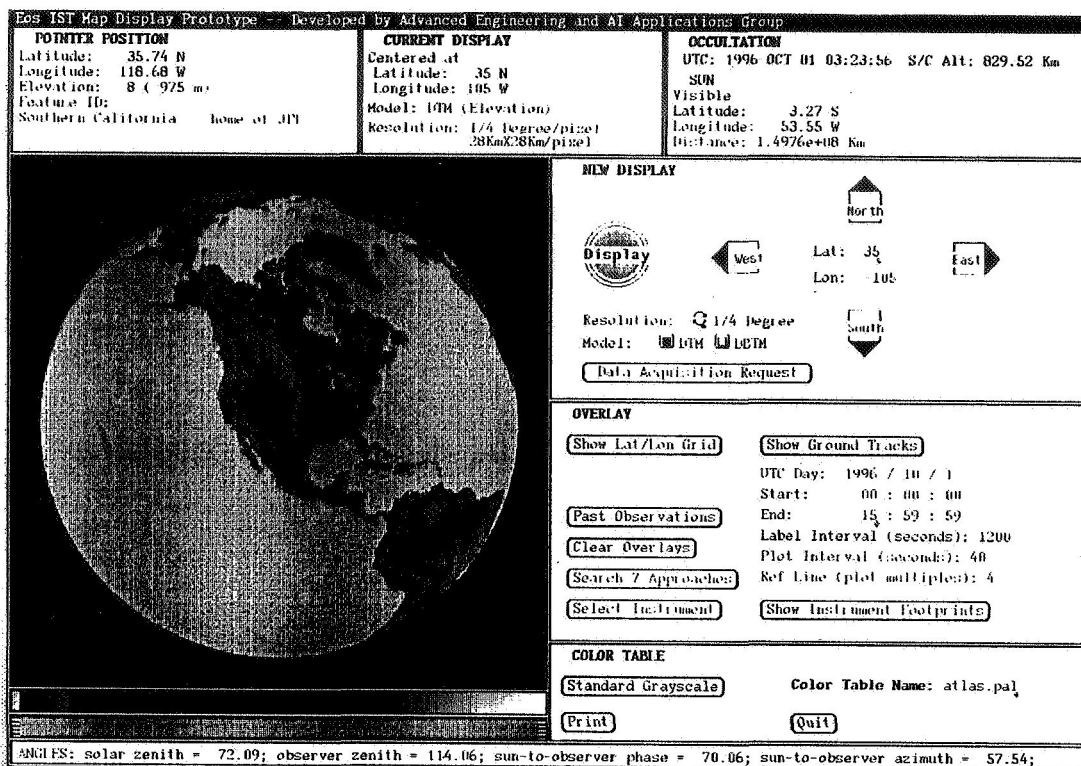


Fig. 3 The EOS Instrument Support Terminal (IST) prototype can be used to visualize spacecraft ground tracks during the planning process. It originally sought to clarify, refine, and discover requirements for the operational IST. In the illustration above, the spacecraft's path is superimposed over the current view of the planet, showing position and orientation for anygiven time.

The security experiment taught us the dangers of being relaxed about security in order not to offend users. Although the External User Access Node's design was sound, our choice of a small PIN size as a user password for the authentication tokens eventually encouraged development of an illicit program that mimicked the token's behavior. No aspect of the system was compromised; but we could no longer be sure active sessions were started by authorized individuals. (Since that experiment, the EUA concept has been replaced by Kerberos and a dedicated router with manually-updated tables, so only trusted hosts from fixed locations can receive packets.)

Some of the SANTA II innovative displays brought praise from the science teams, and are now being improved upon for Galileo's upcoming Earth II encounter. Other lessons from the SANTAs have been incorporated into the SFOC Telemetry Delivery Service (TDS) and have helped improve the stability of the SFOC Galileo adaptation.

MOPSIE

The object of the Mars Observer Pilot SOPC Imple-

mentation Effort (MOPSIE) was to prototype a suite of software, integrated under some sort of graphical user interface, that runs on workstations called SOPCs (Science Operations Planning Computer). The SOPCs are used by planetary exploration mission scientists for several purposes, including monitoring the health and status of the instruments on the MO spacecraft, planning and issuing instrument command sequences based on the computed position of the spacecraft at various times, and exchanging messages and data among themselves and with the SFOC subsystems.

MOPSIE gave us an idea of what effect relatively slow long-distance network communications will have on remote operations. Lessons learned from MOPSIE are especially important as we move from an era of "fly-by" missions to one of long-term orbital missions; scientists (understandably) don't want to remain at JPL throughout the data acquisition phase of an orbiter's mission.

Future Visions

FIST is continuing to keep abreast of new technolo-

gies and prototyping their application with an eye on immediate needs. Among the projects we have lined up in the near future:

- Sequence Verification: Developing a computer model (State Tracker prototype) of a spacecraft's behavior can help in the planning and debugging of command sequences.

- Small Mission Prototype: Numerous smaller missions are likely to become more prevalent in JPL's future and are likely to replace flagship missions like Voyager, Galileo, Cassini. To support the current NASA vision of small missions to each of the planets, FIST is exploring missions operations systems tailored to this new concept.

Our first application of such a system is MESUR (Mars Environmental SURvey) Pathfinder, a mission designed to explore issues relevant to success of a future network mission which aims to place 16 small landers, some or all accompanied by micro-rovers, on the Martian surface.

The MESUR project is considering a wide range of innovations in system development that might cut costs. One such innovation is early integration of the end-to-end data system -- that is, development of flight software in an environment that includes the existing Multimission Ground Data System, to encourage early communication of design decisions among flight and ground software developers.

FIST may be the prototype for the Pathfinder's System Development Laboratory, which will enhance com-

munication among developer by the establishment and use of a single development facility.

- VULCAN (Visualization Utility for Locating Coronal Accelerated Nucleons), a Solar Flare visualization tool employing 3-D modeling software packages to help convert mountains of raw data into scientific understanding.

- Prototypes of small distributed systems built upon OSF's Distributed Computing Environment (DCE), enabling us to announce whether these promising multi-vendor services are mature enough to be built upon. Evaluations are being done with other industry standards (OSF/1, POSIX-compliant system calls, new workstations, and laptop UNIX boxes.

Conclusion

FIST's task of being a "technology gatekeeper" continues to ease new technology into current and future projects within JPL. Having such a prototyping front-end can reduce both risk and development time, while simultaneously ensuring that the products incorporate worthwhile new ideas. Evolutionary prototyping can play a significant role in achieving NASA's new goals of smaller, faster, and cheaper space missions.

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

NASDA'S
ADVANCED ON-LINE SYSTEM
(ADOLIS)

N94-23903

Yoshikatsu Yamamoto, Hideo Hara, and Shigeo Yamada

Tracking and Data Acquisition Department
National Space Development Agency of Japan (NASDA)
Tokyo, JAPAN

Nobuyuki Hirata, Shigenori Komatsu, Seiji Nishihata, and Akio Oniyama

Space Development Division
NEC Corporation
Yokohama, JAPAN

ABSTRACT

Spacecraft operations including ground system operations are generally realized by, large or small, various scale group work which is done by operators, engineers, managers, users and so on, and their positions are geographically distributed in many cases. In face-to-face work environment, it is easy for them to understand each other. However, in distributed work environment which needs communication media, if only using audio, they become estranged from each other and lose interest in and durability of work. It is an obstacle to smooth operation of spacecraft.

NASDA has developed an experimental model of new real-time operation control system called "ADOLIS" (ADvanced On-Line System) adopted to such distributed environment using multi-media system dealing with Character, Figure, Image, Hand-writing, Video and Audio, which is accommodated to operation systems of a wide range including spacecraft and ground system. This paper describes the results of the development of the experimental model.

Key Words: Spacecraft, real-time operation, control system, multi-media, video conference, groupware

1. INTRODUCTION

The ADOLIS is the next generation real-time control system for specifically complex operation required for the future space crafts featured by space experiment, on-orbit unit replacing service, etc.

Generally onboard crew are forced to perform many missions (experiments) within a limited time. Therefore, the most suitable possible operation environment for physical and mental aspects needs to be provided to the crew. NASDA has created environment making it possible on a single workstation to perform many tasks, such as planning coordination, video conferences, automatic checking and monitor display of engineering data, and they have evaluated and identified problems in the operability and technological maturity of such environment.

The evaluation is especially conducted on the following functions.

- (1) Multi-media Interactive Communication

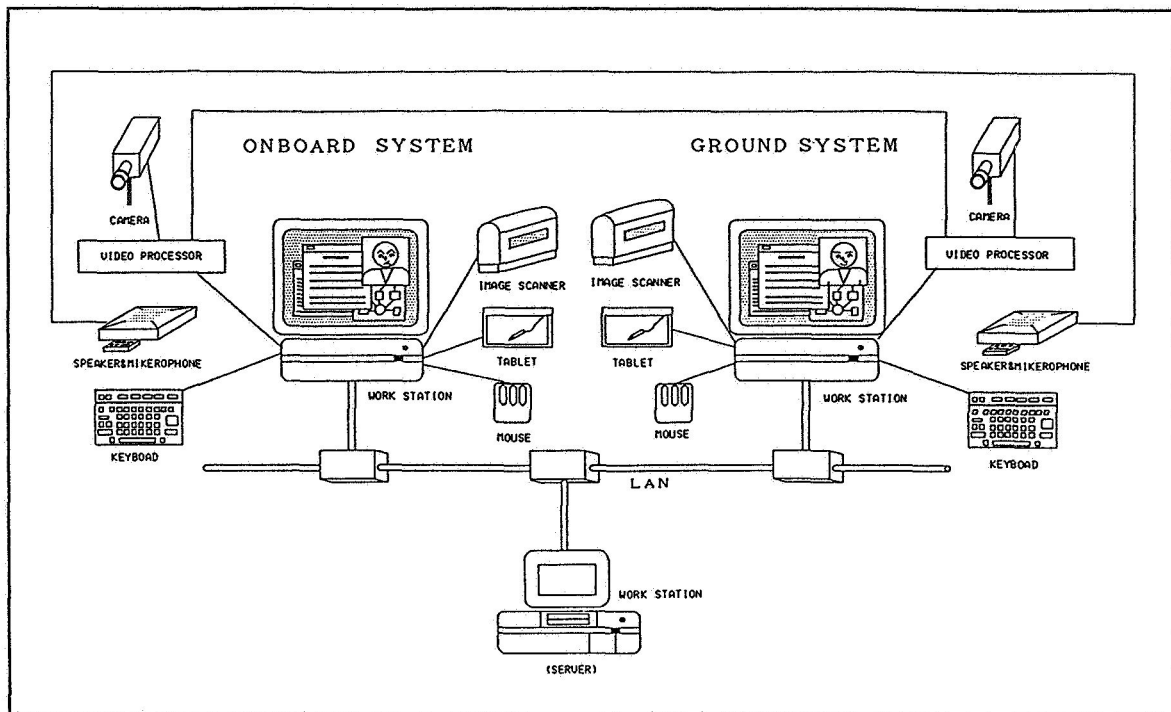


Fig.1 ADOLIS Testbed System Configuration

To simulate operation of space experiment by effective conversation and direction between onboard and ground using multi-media system (video conference system)

(2) Operation Plan and Procedure Management

To simulate coordination of operation plans and procedures and monitoring of the execution between onboard and ground seeing the same information on each screen input from common database.

(3) Contingency Support

To simulate confirmation of trouble occurred onboard, determination of recovery procedure and reconfiguration using semi-automatic support system

(4) Others

- Bilingual operation

Switching between Japanese and English on a screen by a touch

- International standard (Interoperable) human interface and user friendly system
- Security system

2. SYSTEM CONFIGURATION

The ADOLIS testbed system configuration is shown in Fig.1.

2.1 Hardware Configuration

The hardware system of the ADOLIS is composed of three workstations, a server and two terminals (onboard and ground), connected via LAN. Each terminal has a video camera, a video processor, a speaker, a microphone, an image scanner, and a tablet. It has an environment that treats video, graphics, image, handwriting, text on multi-window (X-window). It makes possible visual operation using graphics, etc. in

the environment and smooth understanding between operators by means of video, image, handwriting in addition to audio and text. A terminal system configuration of ADOLIS is shown in Fig.2.

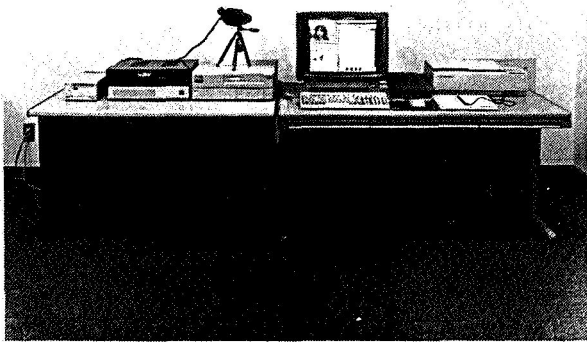


Fig.2 ADOLIS Terminal System Configuration

2.2 Software Configuration

The software configuration of the ADOLIS testbed is as follows.

- Multi-media interactive communication (MERMAID)
- Operation plan and procedure management
- Contingency support
- Telemetry monitoring
- Security management
- Database management
- Basic software (Unix, X-window, etc.)

Unix is used for the operating system of the testbed and a Ostandar d software product, "MERMAID" (Multi-

media Environment for Remote Multiple Attendee Interactive Decision-making), is used for the multi-media interactive communication function.

3. MULTI-MEDIA INTERACTIVE COMMUNICATION FUNCTION

Group work in distributed environment as spacecraft operation needs common space for transmitting understanding and information between the group work members, so called "Groupware", for efficient group work. The ADLIS can treat the multi-media informaion (video, audio, image, handwriting, text) in common space between multiple operation positions.

The screen image of the multimedia interactive communication function is shown in Fig.3. There are two sub-functions in this function as follows.

(1) Video conference function

As operators are looking at the video of other operators on multi-window, they can talk to each other by audio function between multiple positions. They can check the attendees through the video with the image of attendees registered in advance.

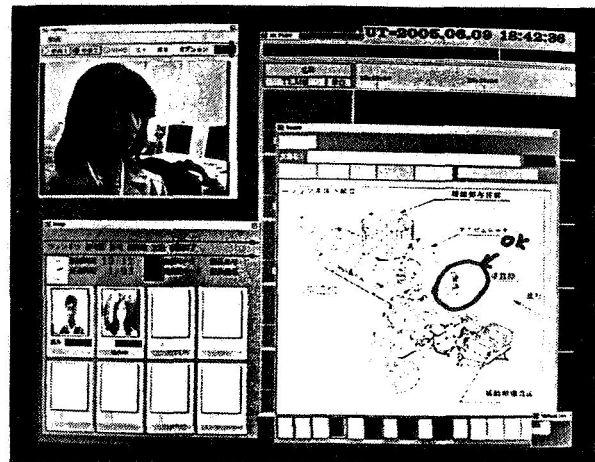


Fig.3 Screen Image of Multi-media Interactive Communication Function

(2) White board function

Operators can input image by scanner and display it on multi-window of each terminal which attends the conference. They can handwrite on the image or white board in multi-window through tablet in real-time.

The following merits are obtained by using this multi-media system.

- Operators can communicate each other based on common understanding and information.
- Holding information in common through common space
- Direction and notification through visual data
- Operators can communicate each other in face-to-face environment.
- Direction and notification through video

Thereofre, this function can improve certainty, reliability and easiness of direction and notification for operation.

4. OPERATION PLAN AND PROCEDURE MANAGEMENT FUNCTION

This function manages master time-line (MTL), detailed time-line (DTL) and operation procedures in database server. Operators can hold them in common. They can coordinate operation plans and procedures and monitor execution of procedures as they are looking at the same information.

This function makes possible paperless operation and can improve promptness and certainty of coordinating and updating plans and procedures. The screen image of the operation plan and procedure management function is shown in Fig.4.

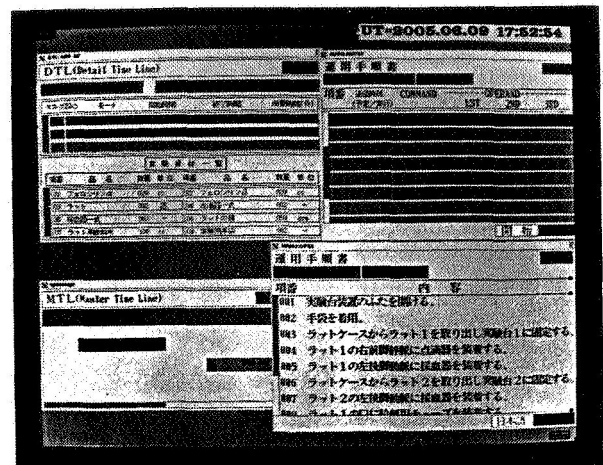


Fig.4 Screen Image of Operation Plan And Procedure Management Function

5. CONTINGENCY SUPPORT FUNCTION

The function supports recovery operation for contingency semi-automatically. When a problem is occurred in a spacecraft, the system block diagram of the spacecraft is displayed and the problem related subsystem is identified by color automatically. When the subsystem is clicked by mouse, more detailed block diagram of the subsystem is displayed and the error component is identified by color.

Some estimated recovery procedures for the problem is also automatically displayed and executed by selection of operator.

The function can improve promptness and certainty of recovery operation and reduce load of operator. The screen image of the contingency support function is shown in Fig.5.

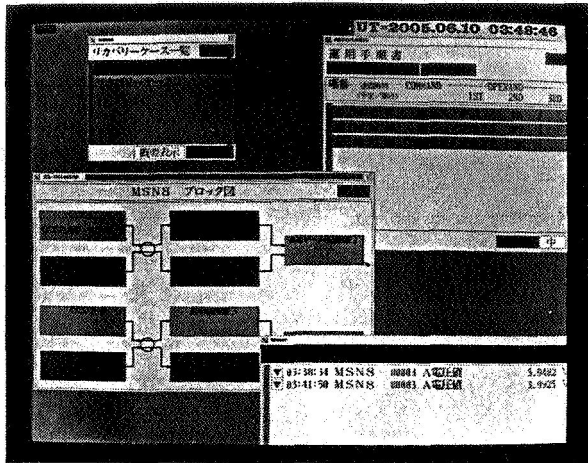


Fig. 5 Screen Image of Contingency Support Function

6. OTHER FUNCTIONS

6.1 Bilingual Operation

As international projects such as the space station program are increasing, this function is supposed to become necessary. At this time, the simple function of switching between Japanese and English by a touch is provided in the operation procedure screen of this testbed. (See Fig. 6) However, in future, an automatic voice translation function may be provided.

6.2 International Standard (Inter-operable) Human Interface And User Friendly System

The international standard human interface is also supposed to become necessary for international project and cross-support operation between agencies. The human interface of the testbed is designed by specialist taking account of the following point, aiming at standard and user friendly system.

- Monitor screen

- Proper quantity of information

on a screen

- Recognizability of status change
- Identification by appropriate color
- Turning his eyes on important part naturally
- Unity and coordination
- Control screen
 - Easy, smooth, intelligible and reliable operation
 - Recovery of operation miss
 - Recognizability of instruction result
 - Appropriate feedback (timing and expression)
 - Attractive and pleasant operation

The screen image of the telemetry monitoring function such as graph, level meter, etc. is shown in Fig. 7.

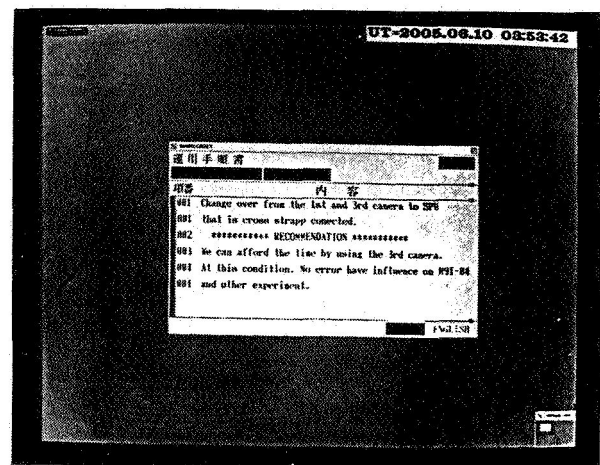


Fig. 6 Screen Image of Bilingual Function

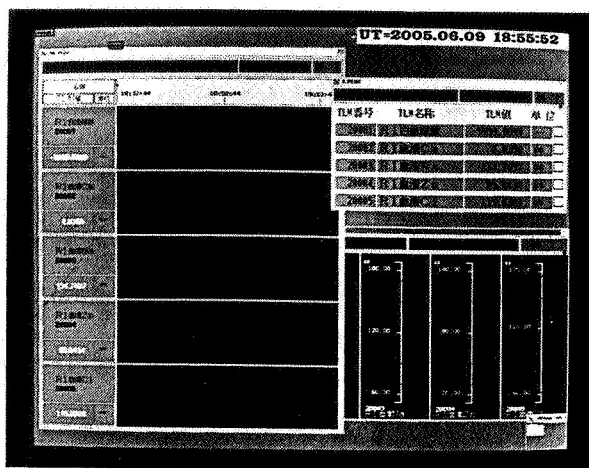


Fig.7 Screen Image of Telemetry Monitoring Function

6.3 Security System

The function establishes access level by kinds of user. For example, operators who permitted to transmit a certain command can access the function of transmitting the command by password.

7. CONCLUSIONS

The results of the experiment are summarized as follows.

- Demonstration of availability of multi-media communication functions through the simulation of spacecraft operation
- Evaluation of the experimental system by NASDA's expert operators and improvement of the system based on the evaluation results
- Human engineering interface design taking account of operability and monitorability by specialist

and the following issues has been identified.

- Remote test using domestic or international communication circuit
- Space link simulation test
- Development of standardized human interface
- Application of the system to a project

It is planned that these study results will be reflected in the operation system for experimental satellite for robot arm and rendezvous-and-docking that NASDA is independently working on and the manned-mission JEM.

8. ACKNOWLEDGEMENTS

The work presented within this paper was performed by many people from NASDA Tracking and Data Acquisition Department(T&DA), Tsukuba Tracking And Control Center(TACC) and NEC Corporation, a contractor of the work.

N94-23904

CONSTRAINT-BASED SCHEDULING

Monte Zweben
 Artificial Intelligence Research Branch
 NASA Ames Research Center
 Moffett Field, CA 94035
 Mail Stop 244-17
 zweben@ptolemy.arc.nasa.gov

ABSTRACT

The GERRY scheduling system developed by NASA Ames with assistance from the Lockheed Space Operations Company, and the Lockheed Artificial Intelligence Center, uses a method called constraint-based iterative repair. Using this technique, one encodes both hard rules and preference criteria into data structures called constraints. GERRY repeatedly attempts to improve schedules by seeking repairs for violated constraints. The system provides a general scheduling framework which is being tested on two NASA applications. The larger of the two is the Space Shuttle Ground Processing problem which entails the scheduling of all the inspection, repair, and maintenance tasks required to prepare the orbiter for flight. The other application involves power allocation for the NASA Ames wind tunnels. Here the system will be used to schedule wind tunnel tests with the goal of minimizing power costs. In this paper, we describe the GERRY system and its application to the Space Shuttle problem. We also speculate as to how the system would be used for manufacturing, transportation, and military problems.

1.0 INTRODUCTION

Efficient scheduling is crucial for manufacturing companies that must balance limited production resources against challenging order requests. Airlines and package delivery companies must schedule large fleets of vehicles coordinating transportation goals with maintenance goals but must also be adaptable to external forces such as weather and equipment failure. The DoD also faces daunting scheduling problems ranging from logistics transport problems to mission planning problems.

NASA also faces complex scheduling problems including telescope observation scheduling, spacecraft crew scheduling, and spacecraft mission planning. Our research is motivated by the Space Shuttle

Ground Processing problem. Ground processing entails the inspection, testing, and repair activities required to prepare a Space Shuttle for launch at the Kennedy Space Center (KSC) in Florida.

This paper describes a scheduling algorithm that is being used to schedule shuttle ground processing but is also applicable to the other scheduling problems alluded to above. First we present our definition of a scheduling problem and then describe our scheduling method. After presenting the general approach we describe how it is used to solve the Space Shuttle problem and then briefly describe how it can be adapted to other real-world problems.

2.0 SCHEDULING

In this section we define the scheduling problem beginning with a simplified version and evolving to a more realistic definition.

2.1 General Problem

Generally scheduling systems are provided a set of activities, relationships between these activities (such as predecessor-successor requirements), resource requirements for each task (i.e., how much of what kind of resources are necessary), and a set of deadlines or milestones. With this input, scheduling systems determine start and end times as well as an assignment of resources to each activity such that: 1) the relationships between tasks are preserved, 2) no resource is over-allocated (i.e., at no time does the demand for a resource exceed its supply), and 3) all milestones are met.

For example, consider a Space Shuttle repair scenario where each Space Shuttle Main Engine needs to be inspected, removed, repaired, re-installed, and tested, in that order. The tasks associated with different engines are unrelated meaning that any task in support of one engine could simultaneously occur with the tasks in support of a different engine. Assume that each task requires 10 technicians, an

engineer, and a safety inspector. Suppose there were only 15 technicians on call for each shift. In this case, no two activities would be able to occur in parallel because together they require 20 technicians and there were only 15 available. If there were more technicians the system would place tasks in parallel in order to meet the milestone.

Consequently, a scheduler would determine activity start times that sequence the activities completely serially because any two activities' demand exceeds the supply of technicians.

In summary, scheduling systems search through the space of possible start times and resource assignments with the goal of finding an assignment that satisfies all domain constraints. These constraints include milestones, resource capacities, and temporal relationships.

2.2 Optimizing and Satisficing

Most scheduling systems simply find an acceptable schedule and then terminate. They are not necessarily concerned with finding the best schedule that satisfies the constraints. In many domains, there is great variability in the quality of schedules that satisfy constraints. For example, an organization might want to find the schedule that uses the minimal amount of overtime labor, or one that minimizes the overall flowtime of a schedule. Unfortunately, deriving the *optimal* schedule is a time consuming process that requires a great deal of combinatoric search. In most cases, near-optimal solutions are sufficient. The process of problem-solving with the goal of finding near-optimal solutions is called *satisficing* [Simon]. The satisficing algorithm presented in the next section continues to search after finding a schedule that merely satisfies constraints, in order to find better quality schedules according to stated optimization criteria.

2.3 State Conditions

Most scheduling systems reason about the changing availability of resources over time but few track the changes of arbitrary conditions. State conditions are attributes of the scheduling problem that change with time. The tasks of a scheduling problem are constrained by these conditions and occasionally the activities change the values of the conditions. Examples include the position of switches and other mechanical parts, the readings of sensors, and the location of objects. Schedulers that handle state conditions must provide a language to specify the additional *state constraints* and to specify the *effects* that tasks have on state conditions.

Examples of state conditions in the Space Shuttle scheduling problem include the position of the payload bay doors, the status of the orbiter's hydraulics system, and whether an area adjacent to the orbiter is hazardous. Examples of state constraints include the rule that no task may take place in a hazardous area. Additionally, some activities require orbiter hydraulics while others require the hydraulics to be off. Likewise, certain activities require the payload bay doors to be in one of their three main positions. Activities also change these conditions. Some activities result in opening or closing doors and turning hydraulics on or off. Similarly, hazardous operations cause the areas surrounding their respective work areas to be considered hazardous thus delaying any other operations that must share those areas.

Our system supports the modeling of state conditions and provides a language for state constraints and task state effects however, details of this language are beyond the scope of this paper. It suffices to say that the satisficing search mechanism presented below considers state constraints and state effects as it schedules.

2.4 Pre-emptive Scheduling

Pre-emption is the process of temporarily suspending activities and resuming them later. Pre-emption can be caused for a number of reasons. In a telescope observation scheduler, the system might interrupt an activity when a more important and rare astronomical event arises. Activities could also be suspended to allow more contentious activities to execute in their limited windows of opportunity. These are examples of *flexible pre-emption*. The Space Shuttle problem requires a more restricted type of pre-emption called *fixed pre-emption*.

Fixed pre-emption is the suspension and resumption of activities according to a strict calendar. In the Shuttle domain the calendar corresponds to work shifts. Some activities can be worked all shifts, every day, while others have certain restrictions such as no weekends, only third shift, or only first shift.

To handle this sort of pre-emption our system requires a calendar for each task that indicates how it is to be pre-empted or split into smaller pieces. For example, suppose a task that requires 12 hours work is assigned a first shift, no weekends calendar. If the task begins at 8:00 A.M. Monday, it will be suspended at 4:00 P.M. that day, then resumed Tuesday morning at 8:00 A.M., and then finally completed at noon. Thus the task spans two calendar days. Suppose however that the task began Friday.

It will then terminate Monday thus spanning four calendar days.

Pre-emption greatly complicates scheduling because of the way it interferes with resources and state conditions. Whenever a new time is considered for a task, the task must be split according to the calendar. However, it is sometimes inappropriate for the state and resource constraints to be valid for the entire period of the pre-empted tasks. For example, the resource needs that correspond to human labor should not be required during the suspended periods of the task's duration. In other words, it makes no sense for employees to be standing around idle. In these cases, the resource and state constraints must be *inherited* to the split tasks thus avoiding the idle periods. Other constraints can remain active throughout the duration of the task. An example of these include a resource request for a heavy piece of equipment that requires significant assembly. The equipment usually remains in the work area, unavailable to others because of the overhead required to set it up.

Our system allows the user to designate which resource requests and which state constraints and effects are to remain valid throughout the suspended period and which ones are valid only during active periods.

3.0 CONSTRAINT-BASED SCHEDULE REPAIR

In this section, we present the search method used by the GERRY scheduling system. The system allows the user to specify a set of *tasks*, a set of *state conditions*, and a set of *resources*.

Tasks have start and end times, resource requests, resource assignments, work durations, and calendars.

Resource pools are defined by the user and have a corresponding maximum capacity. For example, in the Space Shuttle domain there might be a pool of 20 technicians or three pools of 5 forklifts.

State conditions are also provided by the user along with the initial values for each condition. For example, the right-hand payload bay door with an initial value of *closed*.

3.1 Input

- **Task Data** - For each task, the following information is provided:
 - work duration - amount of active work time required for the task to complete.
 - calendar - the pre-emption times for the task.

- resource requests - the list of resource types and quantities necessary.

- **Resource Data** - For each resource pool, the following information is provided:

- type - the name of the resource category that the pool is classified as.
- capacity - the maximum amount of the pool that can be simultaneously assigned.

- **State Condition Data** - For each state condition, the following information is provided:

- initial value - the value for a condition which persists until a task changes it.

3.2 Output

For each task, the following information is determined:

- start time - the beginning of the task.
- end time - the finish of the task.
- resource assignments - the actual resources chosen.

The rules and preferences that schedules must observe are captured by *constraints*. Constraints are relationships that are desired by the user and are composed of the following items:

- **Arguments** - the tasks, resources, or state conditions that are related to each other.

Example: a task and a resource pool are arguments to a resource capacity constraint.

- **Penalty** - a score of how poor the arguments are with respect to the constraint.

Example: if the resource pool argument were overallocated during the time of the task, then the penalty of the resource capacity constraint would be high.

- **Weight** - a number reflecting the importance of the constraint.

Example: resource capacity constraints for scarce resources such as expensive equipment would have higher weights.

- **Repairs** - suggested schedule modifications that are intended to improve the penalty.

Example: move tasks that are involved in an overallocation to a time where more of the resource is available.

Loosely speaking, the penalty is analogous to the amount of money one would have to pay with respect to the current assignment of times and resources. The weight of the constraint reflects its importance when compared to other constraints. Repairs are methods for changing the schedule, either by substituting resources, or by moving, adding, or deleting activities.

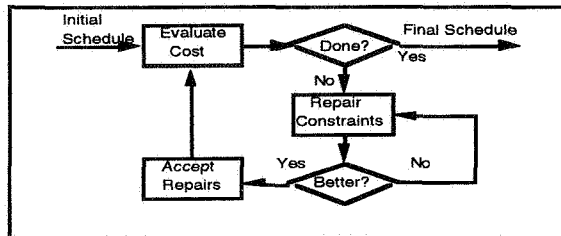


Figure 1: Iterative Repair Scheduling

Figure 1. presents our simple iterative repair algorithm. See [Zweben et. al.][Zweben1 et. al.] for details. The system begins with an initial schedule and then initiates a repair loop.¹ If the problem posed to the system is a rescheduling problem then the initial schedule is the schedule with changes imposed by the user. If the system is scheduling from scratch, then all tasks are placed at their earliest possible start times while preserving temporal constraints. This is accomplished with a well known polynomial (i.e., efficient) algorithm [Davis, Waltz].

In the repair loop, the system calculates the cost of the solution. This calculation is simply the sum of each constraint's penalty multiplied by its weight. If the cost is below a threshold set by the user, the search terminates.² Otherwise, a cross-section of the highly penalized constraints are repaired. We often refer to these constraints as the violated constraints because their penalties exceed a certain threshold.

In short, the system simply starts with a schedule and isolates the violated constraints. Then it moves tasks around and substitutes resources as suggested by the repairs embodied in the violated constraints. It accepts the new schedule if the new cost is lower than the previous cost. If the repaired schedule is worse than previous one, it is rejected and new repairs are

attempted on the previous schedule.³ The system continues this process until the cost of the solution is acceptable to the user, or the system is terminated by the user. The system also terminates if a certain number of iterations have been tried.

¹Similar repair techniques are used in OMP [Biefeld, et. al.] and in the work on the MIN-CONFLICTS heuristic [Minton, et.al.].

²The search also terminates if the system exceeds the iteration bound imposed by the user.

³Actually the system sometimes accepts worse schedules in order to break out of situations called *local minima*. In a local minimum, any repair leads to a worse schedule, but subsequent repairs could improve the schedule so that it is eventually superior. This technique is called simulated-annealing and was originally reported in [Kirkpatrick].

4.0 SYSTEM FUNCTIONALITY

4.1 User Interface Overview

GERRY allows both manual and automatic scheduling thus requiring a sophisticated user interface. The user instructs the interface to display a chart. The chart library is extensible and allows the user to define different views of the schedule. For example, the user could ask for a time-line (i.e., Gantt Chart) and a resource profile (i.e., a histogram of resource usage over time) to be displayed for every task. Alternatively, the user could require a time-line and state condition profile (i.e., a histogram of state conditions over time) for a specific set of tasks. Figure 2. is part of a Space Shuttle schedule with a time-line and resource profile. Figure 3. shows the same schedule (with hourly units) at a coarser level of resolution. Zooming in and out of different levels of resolution is accomplished by clicking in the upper right hand boxes of the chart. This convention was adopted from the COMPASS scheduling system of Barry Fox at McDonnell Douglass in Houston, Texas.

When the chart is displayed each of the activities and histograms are mouse-sensitive. One can drag a task to a new location, modify its status as pending, active, or complete, and ask for a the list of resources that the task uses. The status of an activity is reflected by the shading of the displayed bar. If the activity is shaded black, it is complete. Ongoing activities are outlined in black (but not shaded). Unshaded activities that do not have an outline are pending.

In addition to status, shading is also used to reflect the danger level of the task. If an activity is shaded with a cross-hatch pattern then it is considered hazardous.

The interface supports many task look-up methods. For example, one can scroll to a point in a chart where a particular task begins, one can scroll to an over-allocation, or one could simply use scroll bars.

Also included in the interface are a form editor and a temporal constraint grapher. The form editor, shown in Figure 4., is simply a mechanism to enter new activities and constraints. The grapher, shown in Figure 5., allows the user to inspect the complex temporal relationships between tasks. The grapher works in a demand-driven manner instead of cluttering the display with the entire schedule's graph. One clicks on a task and the graph expands from that point on.

4.2 Schedule Monitoring and Rescheduling

While GERRY can be used as a planning tool for future schedules, its strength is in its ability to monitor schedule execution and adapt to the schedule changes imposed by elements outside the system's control. Users modify tasks by changing their status, dragging them around, changing their constraints, and by adjusting task durations. Users can also add and delete tasks.

One of the most important functions of the user interface is the ability to alert the user to the ramifications of their changes. There are three main charts used to inform the user of what they have done: 1) a before-and-after chart, 2) a constraint violation summary chart, and 3) a constraint violation problem report. The before-and-after chart, shown in Figure 6., reports all the tasks that have been changed by indicating their new position and their old position. The constraint violation summary chart, shown in Figure 7., is a list of the current constraint violations. By clicking on one of the violations in the constraint violation summary chart, a constraint violation problem report appears that explains the conflict. For example, the constraint violation problem report shown in Figure 8. explains why a particular resource capacity constraint is violated. Only the tasks that request the resource during the interval of the violation are displayed and the interval is shaded in color.

After changes are given to the system, the user can manually resolve the outstanding violations or ask the system to use the iterative repair method. When the automated method terminates (or is interrupted) it reports a before-and-after chart. If there are outstanding violations, then a constraint violation summary chart is also displayed.

5.0 PROBLEM DOMAINS

5.1 Space Shuttle Ground Processing

In the Space Shuttle Ground Processing domain we start with schedules provided by an existing project management tool used at the Kennedy Space Center. It uses the critical path method (CPM) to schedule activities at the earliest possible times. These schedules are used by Flow Managers who have the responsibility to prepare the orbiter in time for the designated launch date. The data sets start with about 300-400 activities that expand into thousands of split tasks and constraints. Our project team attends KSC schedule meetings and updates the schedule accordingly. Currently, we are in the process of delivering new schedules to the flow managers and

beginning to use the constraint-based repair method to optimize the schedule. Below we enumerate the constraints used for this application.

The constraints for this application include:

5.1.1. Resource Capacity Constraints

- Arguments:
 - Start of a task
 - End of a task
 - Resource Pool
- Penalty: - Constraint is violated when the pool is over-allocated during the task.
Example: 3 tasks in parallel all need a technician but only 2 are on call.
- Repair:
 - Strategy 1: Substitute
Assign a resource pool of the same type that is not over-allocated.
 - Strategy 2: Move
Move one of the tasks contending for the resource to the next time when there is a sufficient amount of the resource.

To decide which task to move the following heuristics are used:

 - Heuristic 1: Fitness
Prefer to move tasks that use an amount of the resource that is close to the amount over-allocated.
 - Heuristic 2: Slack
Avoid moving tasks that have little slack between their earliest and latest start times.⁴
 - Heuristic 3: Dependents
Avoid moving tasks with temporal dependents (e.g., prerequisites).
 - Heuristic 4: Priority
Avoid moving high priority tasks.
 - Heuristic 5: In Process
Avoid moving tasks that have begun.
 - Heuristic 6: Proximity
Avoid moving tasks that are to begin soon.

⁴Slack time indicates the amount of time a task could slip before it affects the milestone. This measure is calculated from the CPM algorithm mentioned earlier.

5.1.2. State Constraints

- Arguments:
 - Start of a task
 - End of a task
 - State Condition
 - Required State
- Penalty: - Constraint is violated when the condition does not reflect the required state during the task.
Example: The payload bay doors are closed for a task that requires them to be 160 degrees open.
- Repair: - Strategy 1: Move
Move the task to the next time where the state condition reflects the desired state.

5.1.3. Milestone Constraints

- Arguments:
 - End of a task
 - Due Date
- Penalty: - Constraint is violated when the end of the task is completed later than the given date.
- Repair: - Strategy 1: Move
Move the task back earlier, before the given time.

Currently we lack the domain knowledge that would distinguish between the importance of these constraints so they all have the same weight. The system uses these constraints (and their corresponding repairs) to minimize missed launch dates (via milestone constraints) and to minimize over-allocation of KSC personnel (via resource constraints) while maintaining the correct orbiter configurations (via state constraints).

In the near future we intend to include another constraint that demonstrates the flexibility of our system. This new constraint will inform the system to minimize labor costs by avoiding overtime labor on the weekend.

5.1.4. Weekend Constraints

- Arguments:
 - Global constraint.
- Penalty: - Constraint is violated when a large number of tasks intersect the weekend.
- Repair: - Strategy 1: Move

Move the tasks with sufficient slack time off the weekend.

5.2 Manufacturing Problems

In job-shops, there are resources such as machines and human operators. Similar to pre-determined launch dates in the NASA domain, job shops have order due dates. In job shops, each machine has to be set up correctly depending upon the task at hand. Typically jobs follow a process plan that is fairly well known in advance. There are very similar optimization criteria in this domain as there are in the Space Shuttle domain. In fact, the constraints described above are usually applicable. Additional constraints would also be written that would modify the schedule to minimize the number of machine set-ups required thus minimizing flow time. Additionally, constraints that minimize the amount of work-in-process inventory would be incorporated. We claim that a knowledge engineer could easily do this without writing another program but rather simply writing new constraints.

5.3 Airline, Trucking, and Parcel Service Problems

In the transportation sector, large fleets of vehicles must be scheduled on a daily basis. These operations are stricken with unexpected events such as unpredicted malfunctions and malevolent weather. When these events occur, it is crucial to get back on track minimizing impact to the original schedule.

In transportation problems there are additional decision variables that constrain the schedule which include the start and end locations of any task and the speed that one will travel between those locations. Constraints would be added that relate the locations, speed, and duration of the task. Additionally the quantity of certain resource requests must be constrained by the duration. For example, the amount of fuel required by an aircraft is dependent upon how long the plane will travel. Constraints that serve to minimize fuel and delays, while observing safety constraints would be added to the constraints discussed above.

5.4 Military Problems

Many military problems resemble transportation problems but with targeting and probability of success factors added. The tasks are generally trips from one's bases to the enemy's targets (and hopefully back home again). In addition to the transportation constraints discussed above, constraints that model

the appropriateness of various aircraft and ordnances for targets would be required.

5.5 Power Utilization

Ames Research Center is also deploying GERRY to minimize power consumption of the Ames wind tunnels. The rates that local utilities charge NASA are based upon the season, time of day, and quantity of power used. Therefore the wind tunnel test schedule can greatly affect energy costs. Ames will use GERRY to adjust the wind tunnel test schedule to minimize its power costs but maintaining the deadlines imposed by those who need the tunnels. Constraints are used to penalize schedules of high cost while repairs move tasks to lower these costs.

6.0 SUMMARY

We have developed a framework for scheduling called constraint-based iterative repair. This framework supports complex scheduling problems where satisficing is required. GERRY, the system based upon this framework, is operational and is being deployed at the Kennedy Space Center in Florida in support of Space Shuttle Ground Processing. The system uses the optimization criteria encoded as constraints to find near-optimal schedules. We claim that our approach is amenable to other problems faced within industry and government and welcome others to apply it.

ACKNOWLEDGEMENTS

Thanks to the entire GERRY project team at NASA Ames, the Lockheed Space Operations Company, and the Lockheed Artificial Intelligence Center. Also thanks to Martin Cohen for his careful review of this paper.

BIBLIOGRAPHY

- [Biefeld, et. al.] Biefeld, E., and Cooper, L., Bottleneck Identification Using Process Chronologies, In *Proceedings of IJCAI-91*, 1991.
- [Davis] Davis, E., Constraint Propagation with Interval Labels, *Artificial Intelligence*, 32(4), 1987.
- [Kirkpatrick] Kirkpatrick, S., Gelatt Jr., C., Vecchi, M., Optimization by Simulated Annealing, *Science*, 220, 1983
- [Minton, et. al.] Minton, S., Philips, A., Johnston, M., Laird, P., Solving Large Scale CSP

and Scheduling Problems with a
Heuristic Repair Method, In *Proceedings
of AAAI-90*.

[Simon] Simon, H., *The Sciences
of the Artificial*, MIT Press, 1969

[Waltz] Waltz, D. Understanding Line
Drawings of Scenes with Shadows,
In P. Winston, editor, *The
Psychology of Computer Vision*,
McGraw-Hill 1975.

[Zweben, et.al.] Zweben, M., Deale, M., and
Gargan, B., Anytime Rescheduling,
In *Proceedings of the
DARPA Workshop on Innovative
Approaches to Planning
and Scheduling*, 1990.

[Zweben1 et.al.] Zweben, M., Deale, M., and
Gargan, B., An Empirical Study of
Rescheduling Using
Constraint-Based Simulated Annealing, In
*Proceedings of the IJCAI-
91 Workshop on Production Planning and
Scheduling*, 1991.

CUSTOMIZING GRAPHICAL USER INTERFACE TECHNOLOGY FOR SPACECRAFT CONTROL CENTERS

N94-23905

Edward Beach and Peter Giancola
Computer Sciences Corporation
Laurel, Maryland, U.S.A

Steven Gibson
Integral Systems, Inc.
Lanham, Maryland, U.S.A

Ronald Mahmot
NASA Goddard Space Flight Center
Greenbelt, Maryland, U.S.A

ABSTRACT

The Transportable Payload Operations Control Center (TPOCC) project is applying the latest in graphical user interface technology to the spacecraft control center environment. This project of the Mission Operations Division's (MOD) Control Center Systems Branch (CCSB) at NASA Goddard Space Flight Center (GSFC) has developed an architecture for control centers which makes use of a distributed processing approach and the latest in Unix workstation technology. The TPOCC project is committed to following industry standards and using commercial off-the-shelf (COTS) hardware and software components wherever possible to reduce development costs and to improve operational support. TPOCC's most successful use of commercial software products and standards has been in the development of its graphical user interface. This paper describes TPOCC's successful use and customization of four separate layers of commercial software products to create a flexible and powerful user interface that is uniquely suited to spacecraft monitoring and control.

1. BACKGROUND

The TPOCC project was initiated in 1985 by the Control Center Systems Branch as a research and development effort. The goal was to establish an architecture that would enhance the capabilities of spacecraft control centers while simultaneously reducing the per mission cost of implementation. Through a series of prototypes, an architecture was chosen based on the following elements:

- a distributed processing approach
- industry standards
- COTS hardware and software components
- reusable custom software

The first operational system developed by TPOCC was initiated in 1989 as a replacement for the existing control center supporting the International Cometary Explorer (ICE) and the International Monitoring Platform (IMP). TPOCC successfully supported its first launch of a new spacecraft with the deployment of the Solar Anomalous and Magnetospheric Particle Explorer (SAMPEX) in July 1992. SAMPEX's users represent a special challenge for the TPOCC user interface because no SAMPEX spacecraft contact lasts more than 10 minutes. These users need the ability to rapidly access all available data, automate their activities, and quickly respond to contingencies. Other new missions have also adopted TPOCC's architecture and software as shown below.

ICE	International Cometary Explorer
IMP	International Monitoring Platform
ISTP Series	Polar Plasma Laboratory (POLAR); Solar and Heliospheric Observatory (SOHO); Interplanetary Physics Laboratory (WIND)
SMEX Series	Fast Auroral Snapshot (FAST); Solar Anomalous & Magnetospheric Particle Explorer (SAMPEX); Submillimeter Wave Astronomy (SWAS)
TRMM	Tropical Rainfall Measurement Mission
XTE	X-Ray Timing Explorer

*Satellites Employing the TPOCC
Architecture*

TPOCC's commitment to a standards-based approach led to the selection of the X window system from the Massachusetts Institute of Technology (MIT) as the core of the TPOCC display subsystem. Three other layers of commercial software complement the X protocol and complete the TPOCC user interface capabilities. The complete history of the TPOCC display subsystem is shown below, starting with the early work evaluating various windowing systems (X, NEWS, and SunView) and leading up to the present design that includes COTS products such as Hewlett Packard's (HP's) Visual User Environment (VUE) and the user interface design tool Builder Xcessory. All of the milestones listed in the table are explained in detail in the sections that follow.

1988	NEWS evaluated for possible use Prototype built using SunView
1989	Use of X Windows initiated (X11R3) Prototype using Athena widget set
1990	Prototype ported to Motif widget set UIL becomes display definition language
1991	TPOCC page editor written Port to X11R4 and Motif 1.1.3 Graphical widgets added (dials and plots)
1992	HP VUE integrated SAMPEX launched Command panel written Builder Xcessory integrated

History of the TPOCC Display Subsystem

The overriding philosophy behind TPOCC's display design is that close adherence to the leading open systems standards will provide tremendous benefits. The importance of this tenet is highlighted in the sections that follow. The standard products selected by TPOCC have many significant features, such as:

- hardware independence
- remote display distribution
- support for object-oriented software design
- extensible widget set
- extensible display definition language

But in addition to these features, a strict policy of standards adherence provides benefits of its own. It allows rapid integration of COTS products based on the same standards. It has also made it possible for TPOCC to jointly develop a common library of display objects with projects in different NASA organizations. Finally, it allows external

organizations to create graphical applications that fit into the TPOCC user environment with little or no modification.

2. THE X WINDOW SYSTEM

The distributed environment used by the TPOCC architecture necessitates the use of a network based windowing system. The X Window System was chosen to serve as the basis for TPOCC's graphical user interface because of its wide acceptance and superior performance. The other alternatives available at the time (late 1988) were either slow and not widely accepted (NEWS), or vendor-specific (such as SunView). In 1989, TPOCC made a full-scale commitment to X and began using the software shipped with MIT's X Version 11 Release 3 as the lowest layer supporting the rest of TPOCC's user interface capabilities.

X Window's client/server model layered on networking standards such as Ethernet and Transmission Control Protocol (TCP)/ Internet Protocol (IP) is identical to the distributed processing techniques used throughout the TPOCC architecture. The benefits of this client/server model include:

1. hardware independence: any terminal supporting the X protocol can display data, regardless of graphics hardware
2. remote display distribution: displays can be sent to remote locations on the Ethernet without modifying software
3. reduced cost per user position: the ability to use X-terminals reduces the cost of a single user position to less than half the cost of a workstation

MIT supplies the "X lib" software library to support the client side of this model and to provide routines that can be called to build proper X protocol messages. However, another library layer exists in X, namely the X toolkit. This collection of routines support an object-oriented paradigm in a traditional programming language (C). The toolkit allows the creation of sets of display objects, called widgets, that exhibit object-oriented properties such as inheritance, data hiding, and polymorphism. TPOCC has taken advantage of the power of the toolkit to create their own custom widget set that is tailored to a real-time control center environment. Furthermore, any widget written according to the X toolkit standard can be easily shared by another project following the same approach. TPOCC is currently exchanging widgets with several projects including Goddard's Data Systems Technology Division's Generic Spacecraft

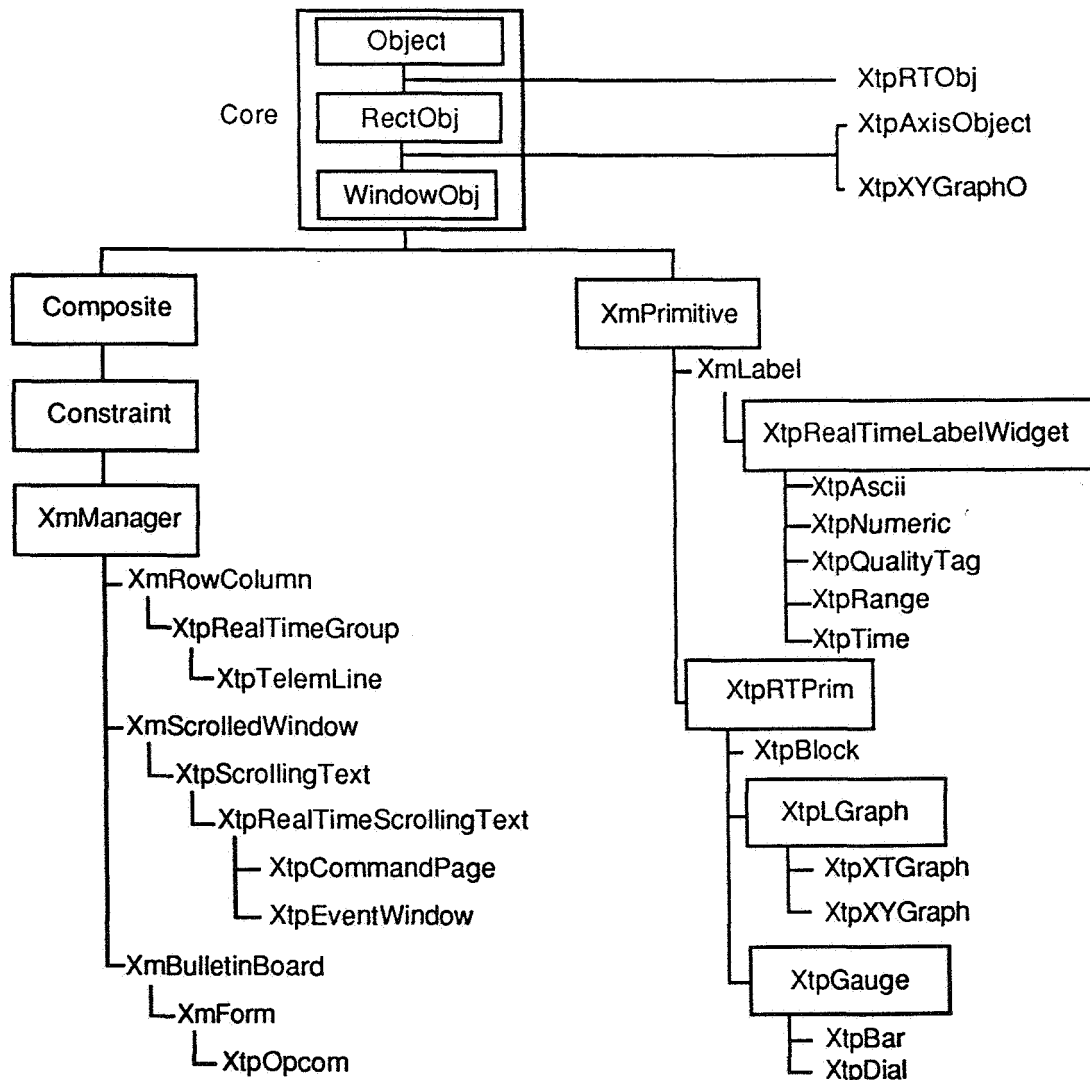
Analyst Assistant (GenSAA) project. In effect, the projects are creating a common pool of display capabilities that can be developed in pieces but rapidly merged when an operational system is being put together.

3. OSF/MOTIF

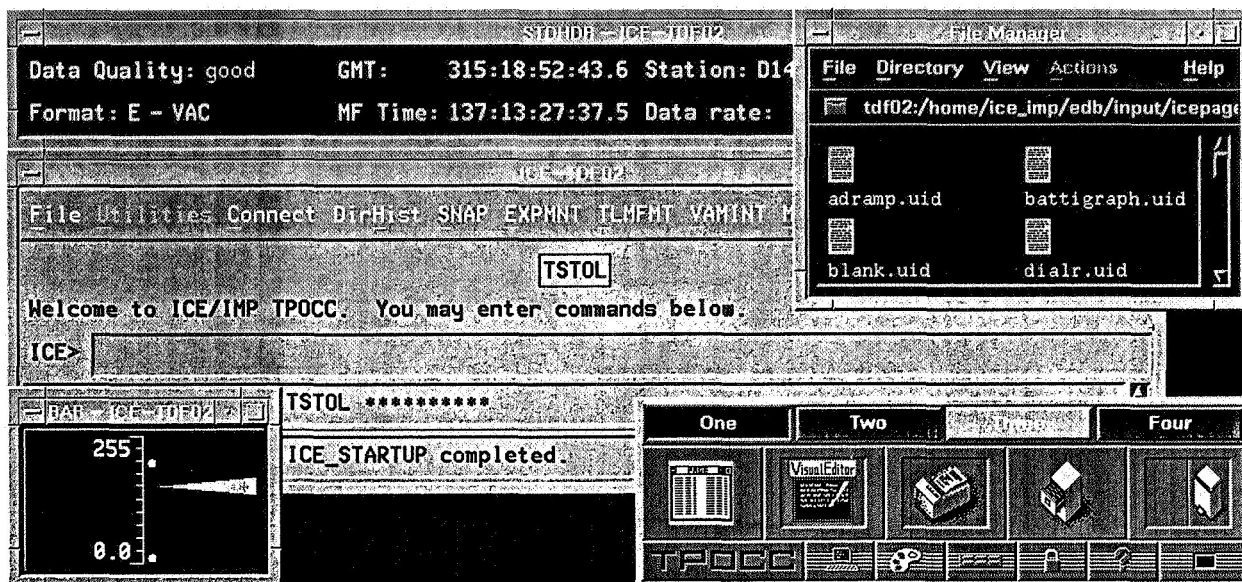
TPOCC's earliest widget set was an extension of the Athena widget set supplied with MIT's X window software distributions. The publically-available Athena widgets were used as containers to hold individual display objects and as labels and annotations on TPOCC's displays. The TPOCC widget set inherited properties of some of the existing Athena widgets to simplify coding. However, the Athena widget set had numerous shortcomings at the time, principally because it had been written more as

an example of X toolkit coding techniques than as a viable commercial product. TPOCC began to look elsewhere for a more robust widget set, preferably the one emerging as the industry standard. Selecting a widely used widget set would ensure that all TPOCC applications exhibited a standard "look and feel" that would be consistent with any commercial software tools that could be integrated into the TPOCC environment.

So, the Open Software Foundation's (OSF's) Motif user environment was selected, because it enjoyed the widest support in the Unix community and for its extensible user interface language (UIL) discussed below. TPOCC's widget set was ported to Motif in 1990 and now, in 1992, includes over 20 widgets (identified by TPOCC's "Xtp" prefix below).



TPOCC Widget Hierarchy



TPOCC's HP VUE Window Environment

a fully graphical interface that does not require the knowledge of any low-level command languages (such as the Unix shells and UIL). In addition, the TPOCC project has customized HP VUE to translate many control center operations into graphical sequences involving buttons, menus, and icons. As an example, all TPOCC applications are invoked through VUE icon selections. Similarly, files can be edited, printed (in various styles according to the POCC file type), or deleted just by dragging their icon to the editor, printer, or trash can drop target.

Perhaps the most powerful feature of HP VUE is the concept of workspaces. A workspace is a logical grouping of windows that allows the user to view just a single group at a given time. Once each window has been brought up the user can assign it to any workspace through a simple menu operation. Switching between workspaces is as simple as pushing the proper button on the HP VUE workspace manager panel (located at the bottom right of a TPOCC display as shown on the screen above). This capability proved to be especially useful in support of the SAMPEX launch. During launch and early orbit checkout, up to 32 users were using the TPOCC system, with each user viewing from 5 to 10 different pages. Since there is far more information on this number of pages that can be viewed on a single console, the ability to group the windows proved invaluable. Each user would define four workspaces and would cycle through to each group as necessary.

6. FUTURE DIRECTIONS

Although TPOCC's graphical user interface has come a long way in the last four years, active work

continues. New widgets to support schematic displays are being built in conjunction with GenSAA. The widget set is being modified to support data from any arbitrary source, not just the TPOCC data server process. Motif's new support for drag-and-drop operations will be incorporated. Several new applications are being developed, including definition file editors and a graphical debugger for the TPOCC System Test and Operations Language (TSTOL). Improved support for TSTOL itself enhances the user interface, for TSTOL complements the display software by allowing sequences of user activities (such as page selection and placement) to be automated through a set of commands known as a TSTOL procedure. Other capabilities will be added to the reusable software base as requirements are received from new missions. However, the goals established when the TPOCC display development began appear to have been met. A powerful, efficient user interface that improves operational capabilities has been created at minimal cost by maximizing the use of commercial software. Just as importantly, the TPOCC display subsystem has achieved complete reusability from one mission to the next. The display software is one of TPOCC's few subsystems that does not require mission-unique code to support different spacecraft. Each mission defines unique displays in UIL, but the costly process of creating or modifying software for each mission is eliminated.

7. REFERENCES

1. Mahmot, Ronald and Beach, Edward. 1992. TPOCC Graphical User Interface for Editing Display Pages. 1991 *Research and Technology Report*, Goddard Space Flight Center, 208-211.


```

!@(#) File name: dial.uil Release: 1.9
module dial_page
names = case_sensitive
include file 'XmAppl.uil';
include file 'XtpAppl.uil';
value line_offset: 20;
object
  XtpTopLevel: XmForm
  { arguments
  {
  };
  controls
  {
    XmLabel page_title;
    XtpDial dial1;
  };
};
page_title: XmLabel
{ arguments
{
  XmNtopAttachment = XmATTACH_FORM;
  XmNleftAttachment = XmATTACH_FORM;
  XmNrightAttachment = XmATTACH_FORM;
  XmNleftOffset = 5;
  XmNrightOffset = 5;
  XmNlabelString = compound_string(
    '- XtpDial Widget Example -', separate=true)
    & '(absolute resize)';
};
};
dial1 : XtpDial
{ arguments
{
  ! XmForm Constraint resources
  XmNtopAttachment = XmATTACH_WIDGET;
  XmNtopWidget = page_title;
  XmNtopOffset = line_offset;
  XmNleftAttachment = XmATTACH_FORM;
  XmNrightAttachment = XmATTACH_FORM;
  XmNbottomAttachment = XmATTACH_FORM;
  ! XtpDial resources - XmPrimitive Class Part
  XmNborderWidth = 2;
  ! XtpDial resources - XtpRTPrim Class Part
  XtpNmnemonic = default_cvt_ul1;
  XtpNprocess = default_decom;
  XtpNvariableType = XtpCVT_UL1;
  ! XtpDial resources - XtpGauge Class Part
  XtpNscaleMax = "255";
  XtpNshowMinmax = true;
  ! XtpDial resources - XtpDial Class Part
  XmNmargin = 4;
  XtpNdrawPost = true;
};
};
end module;

```

UIL Definition for TPOCC Page "DIAL"

UIL is a powerful language that supports all possible combinations of Motif-based widgets. However, one lesson learned during the TPOCC effort has been that end user creation of display windows directly in UIL is too time consuming and requires detailed

knowledge of the Motif widget set. With this problem in mind, the TPOCC page editor was created. This application, written to comply with Motif's Style Guide, allows users to rapidly design pages using a row-column layout. About eighty percent of control center telemetry displays have been traditionally formatted this way. The intuitive Motif interface provided by the page editor has reduced the time required to construct a page from several hours to about twenty minutes (Ref. 1).

4. BUILDER XCESSORY

However, the page editor does not solve the problem of laying out displays with merged text and graphics. The graphical elements of the TPOCC widget set necessitate a drag-and-drop style layout and design tool. TPOCC therefore began a search for a user interface management system (UIMS) that allows user created widgets to be incorporated into the tool. Builder Xcessory (BX) from Integrated Computer Solutions is a UIMS that uses UIL's WML facility to define new widgets for the BX palette. Thus, TPOCC can create a single WML definition file and come up with both an extended UIL compiler and an extended design tool. TPOCC is in the final phase of integration with Builder Xcessory and expects to be installing this product in the operational TPOCC control centers in early 1993.

Although the primary purpose for integrating Builder Xcessory is to allow users to develop more complex pages than those supported by the page editor, this tool is also quite useful to TPOCC software developers. In this domain, BX is used to prototype screen designs for new Motif applications. Before writing any code, TPOCC developers demonstrate the BX "mock-up" of the future application to gain valuable ideas and feedback from TPOCC users. The command panel, an application that lets flight operations team members plan and conduct the activities for each spacecraft contact through a simple button interface, was prototyped in this manner. Two rounds of demonstrations were held before a final design was reached. The early feedback possible with such a methodology helps reduce the life cycle costs of developing this type of application; prototyping is now a standard part of developing a new application for TPOCC.

5. HP VUE

Finally, TPOCC is using Hewlett Packard's Visual User Environment (HP VUE), the fourth and final layer of COTS software, to provide a graphical interface to the Unix operating system. Shielding the user from Unix's command language is a significant step toward TPOCC's goal of creating

3.1 TPOCC Widget Set

The real-time telemetry display widgets supported by TPOCC can be divided into three types, with each type supporting a different form of data representation. A fourth type, schematic widgets, will be added in the near future and is discussed in the final section below.

3.1.1 Textual Widgets

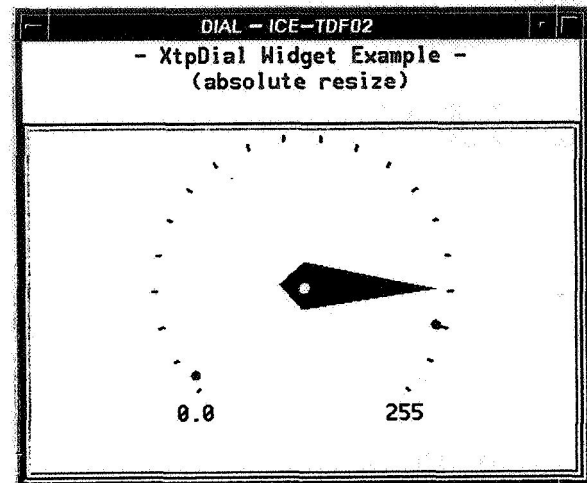
The textual widgets in the TPOCC widget set are divided into two types: those based on Motif's Label widget and those based on Motif's List widget. TPOCC's label widgets are used to animate different types of real-time data at a fixed screen location. Widgets of this type include XtpAscii, XtpNumeric, XtpRange, and XtpTime. TPOCC's list widgets are used to animate a scrolling display of lines of text. Widgets of this type include XtpCpage and XtpEvent.

3.1.2 Gauge Widgets

The first graphical widgets implemented for the TPOCC widget library were gauge widgets. Currently, two TPOCC gauges exist, XtpBar and XtpDial. An XtpDial widget is shown at right. The gauge widgets animate based on a single telemetry point and show not only the current value of the parameter but also its minimum and maximum values during the current spacecraft contact.

3.1.3 Plot Widgets

The most sophisticated widgets are the plot widgets. TPOCC currently supports two such widgets: XtpXTGraph and XtpXYGraph. XtpXTGraph stripcharts a single parameter against time and scrolls based upon a user-selected policy. XtpXYGraph can plot up to five parameters against either a sixth parameter or time. Two independent Y scales are

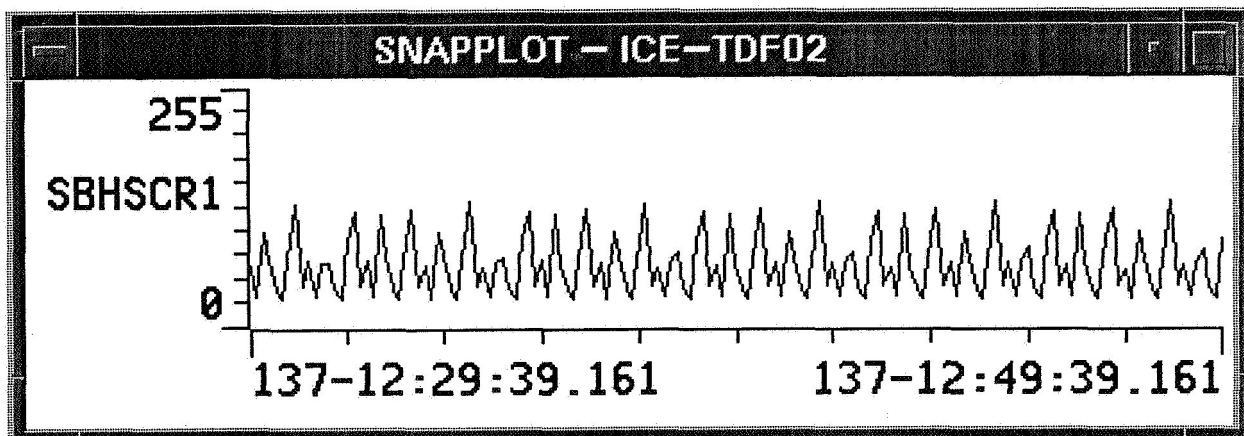


TPOCC's Dial Widget

supported, but the X axis is fixed (scrolling is not supported). An XtpXYGraph widget is shown receiving real-time data below.

3.2 TPOCC's Use of UIL

Motif's UIL plays a key role in TPOCC's display design. Not only does UIL provide the ability to augment the Motif widget set through OSF's Widget Meta Language (WML), but it also allows unlimited flexibility in page design and pages to be created without rebuilding any software. Through WML, the UIL language can be expanded to include user-defined widgets such as the real-time set developed by TPOCC. When WML is run, a new UIL compiler is generated that accepts specifications for TPOCC widgets as valid syntax. A TPOCC UIL fragment is shown on the following page. This UIL defines a single control center display window, known as a page. The definition is for the TPOCC Dial widget window shown in the figure above.



TPOCC's X-Y Graph Widget

SESSION G

DATA ACQUISITION

CO-CHAIRPERSONS

R. W. Burt, JPL, California Institute of Technology, USA
P. Maldari, ESA European Space Operations Centre, Germany

Summary	493
P. Maldari	
G.1 Applications of CCSDS Recommendations to Integrated Ground Data Systems (IGDS)	495
H. Mizuta, NASDA, Tokyo, Japan; D. Martin, H. Kato, and H. Ihara, Hitachi Ltd., Space Systems Division, Yokohama, Japan	
G.2 The Advanced Orbiting Systems Testbed Program: Results to Date	501
P. A. Newsome and J. F. Otranto, CTA Incorporated, Rockville, Maryland, USA	
G.3 Reliable Transfer of Data from Ground to Space	507
F. Brosi, CTA Incorporated, Rockville, Maryland, USA	
G.4 The EURECA Telecommanding Chain: Experience with Packet Telecommand and Telemetry Systems	513
C. Müller and R. Bater, Logica Space and Communications Ltd., London, UK; E. M. Sorensen, ESA European Space Operations Centre, Darmstadt, Germany	
G.5 Verification of Technical Elements of the Advanced Spacecraft Based Upon the CCSDS Recommendation	519
H. Hara, S. Yamada, and Y. Yamamoto, NASDA, Tokyo, Japan; S. Fujii, K. Tumura, M. Yokomizo, and H. Ogasawara, Fujitsu Ltd., Tokyo, Japan	
G.6 Distributed Computing Environments for Future Space Control Systems	525
P. Viallefont, CNES Toulouse Space Centre, France	
G.7 Digital Test Signal Generation: An Accurate SNR Calibration Approach for the DSN	531
B. O. Gutiérrez-Luaces, JPL, California Institute of Technology, Pasadena, California, USA	
G.8 Automatic Tools for System Testing	537
N. M. Peccia, ESA European Space Operations Centre, Darmstadt, Germany	
G.9 Representing Operations Procedures Using Temporal Dependency Networks	541
K. E. Fayyad and L. P. Cooper, JPL, California Institute of Technology, Pasadena, California, USA	
G.10 Situation Management in the Link Monitor and Control Operator Assistant (LMCOA)	547
R. W. Hill, Jr., and L. F. Lee, JPL, California Institute of Technology, Pasadena, California, USA	

G.11 Using Mach Threads to Control DSN Operational Sequences	553
J. Urista, JPL, California Institute of Technology, Pasadena, California, USA	
G.12 Deep Space Network (DSN), Network Operations Control Center (NOCC) Computer-Human Interfaces	561
A. Ellman and M. Carlton, JPL, California Institute of Technology, Pasadena, California, USA	
G.13 A Method for Interference Mitigation in Space Communications Scheduling	565
Y. F. Wong and J. L. Rash, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA	

SUMMARY — SESSION G

P. MALDARI

European Space Operations Centre

The title of this session is open to a wide interpretation, since all engineering activities for space research and/or space applications are related to the generation, transfer, and interpretation of data, therefore implying a data-acquisition process.

The session co-chairpersons realised this at the beginning of the paper-selection process when noting the different topics treated in the submitted papers. Selecting papers was not an easy task in view of this factor and the need, dictated by the limitation of time available for public presentation, to select a limited number of papers out of a large number of excellent submissions.

The session was articulated in five main topics, listed below.

CCSDS-RELATED TOPICS AND PROTOCOLS

Papers G.1 to G.5 were presented during this part of the session. The focus has been on CCSDS/AOS recommendations and related prototyping and testbed activities performed worldwide by major organisations such as NASA, ESA, NASDA, etc. Testbed activities are considered very important for the promotion of standardisation of interfaces between different infrastructures belonging to different agencies and consequently for the maximisation of utilisation of existing ground resources in support of future complex missions requiring international cooperation.

It is believed that today's CCSDS recommendations represent tomorrow's technology on space applications. Consequently, it is recommended that we closely monitor and promote progress in CCSDS prototyping and testbed activities. As a conclusion to this part of the session, it was recommended that a CCSDS session be included in future SPACEOps symposia.

TECHNICAL TRENDS

The paper G.6 by Mr. Viallefont (CNES, Toulouse) summarised technical trends in space control systems (distributed computer architecture and object-oriented processing) and provided challenging ideas for further exploitation, such as virtual computers and fault-tolerant system architecture.

TESTING TOOLS

Testing has also been considered as part of data acquisition, since its purpose is to ensure that the performance of the system meets the requirements and consequently data is acquired as specified.

Two very interesting papers (G.7 and G.8) were presented during the session, the first dealing with a new method of accurately simulating signal-to-noise ratio for calibration of stations in support of deep space missions, and the second reporting on an ongoing feasibility study for automating spacecraft control system testing and on a related prototype being built on commercial software to automatically test some basic functions of spacecraft control.

Testing is a very manpower-intensive activity, but it is essential in setting up a space system. As stressed during this subsession, some efforts are still required to improve testing tools so as to render the mission-preparation phase more ergonomic.

ARTIFICIAL INTELLIGENCE

Three interrelated papers (G.9, G.10, and G.11), dealing with the problematics of configuration, calibration, testing, and operation of deep space links between spacecraft and control centre, were presented during this subsession. The solution adopted and its architecture as applied to the Deep Space Network (DSN) were presented. The porting of the developed technology to non-deep space missions was recommended as a further step towards generic mission planning.

COMPUTER-HUMAN INTERFACE

Paper G.12 presented the computer-human interface (CHI) solutions adopted in the new Network Operations Control Center installed in 1991 for DSN control. The presentation was complementary to a demonstration in the Pasadena Center, where the symposium was held. As stressed during the presentation, CHI technology is a key element of successful space operations. The session was well attended and very animated, with questions and answers particularly in the areas of CCSDS-related topics, technical trends, and testing tools.

N94-23906

APPLICATIONS OF CCSDS RECOMMENDATIONS TO INTEGRATED GROUND DATA SYSTEMS (IGDS)

Hiroshi Mizuta^A, Daniel Martin^B, Hatsuhiko Kato^B, and
Hirokazu Ihara^B

^ANational Space Development
Agency of Japan
World Trade Center Building
2-4-1 Hamamatsu-Cho
Minato-Ku, Tokyo 105 Japan
TEL: 81-3-5470-4255
FAX: 81-3-3432-3969

^BSpace Systems Division
Hitachi, Ltd.
216 Totsuka
Yokohama 244 Japan
TEL: 81-45-865-7027
FAX: 81-45-864-5007

^CSpace Systems Division
Hitachi, Ltd.
4-6 Kanda-Surugadai
Chiyoda-ku, Tokyo 101 Japan
TEL: 81-3-3258-1111
FAX: 81-3-3258-5496

ABSTRACT

This paper describes an application of the CCSDS Principle Network (CPN) service model to communications network elements of a postulated Integrated Ground Data System (IGDS). Functions are drawn principally from COSMICS, an integrated space control infrastructure (Ref. 1), and the Earth Observing System Data and Information System (EOSDIS) Core System (ECS) (Ref. 2). From functional requirements, this paper derives a set of five communications network partitions which, taken together, support proposed space control infrastructures and data distribution systems. Our functional analysis indicates that the five network partitions derived in this paper should effectively interconnect the users, centers, processors, and other architectural elements of an IGDS. This paper illustrates a useful application of the CCSDS Recommendations to ground data system development.

Key Words: COSMICS, CCSDS, IGDS, EOSDIS, ECS, networks

1. BACKGROUND

In 1990, at the 2nd AIAA/NASA International Symposium on Space Information Systems, some of us proposed a unified information and control infrastructure for the coming space age. (Ref. 3) That paper described a system concept, referred to as the Cosmic Information and Control System (COSMICS), which integrates the necessary off line processing and support functions with the on line elements needed for aerospace craft control. The authors stated that a unified control facility might: (a) Minimize life cycle costs, (b) Pro-

vide for technological innovation and support evolutionary changes in social needs, (c) Evolve economically through cooperative and well placed additions of processors, communications, and control facilities, and (d) Make the latest technology available to the widest distribution of users in the most timely fashion.

The tasks required for control of spacecraft operations discussed in the COSMICS paper are similar to, and can be grouped into the same functional categories as, those identified for the Flight Operations Segment (FOS) of NASA's ECS. (Ref. 4) We distributed the combined functionality of COSMICS and the ECS segments across a network architecture comprised of the following architectural elements: 1) A user community which requires data services, 2) A mission operations element to control aerospace craft, 3) An element to accomplish information processing and distribution functions, 4) Governments, with both national and international relationships, and 5) Industry.

Space data systems are being designed to implement standardized and internationally agreed upon techniques of data handling, data classification, and data transmission. (Ref. 5) The Consultative Committee for Space Data Systems (CCSDS) has published recommendations in the following major areas:

1) Architectural specifications for the CCSDS Principle Network (CPN). These define a conceptual model for data handling networks to provide end-to-end data services in support of space missions. The CPN consists of an onboard network which resides on a spacecraft capable of orbit, a

ground network which communicates with the spacecraft, and a space link subnetwork for connectivity. (Ref. 6)

2) Recommendations dealing with techniques for telecommand and telemetry (Ref. 7,8), and

3) Specifications for standardized data units, as well as for time code formats, radiometric and orbit data, radio frequency and modulation systems, ground networks, and other topics.

Recent work by the CCSDS has centered on developing recommendations to make possible cross-support services. (Ref. 9) As defined, this class of services will enable space agencies to bidirectionally transfer data from other space agencies between ground and space systems. (Ref. 10) It has the potential, through standardized implementations of data units and communications services to make possible apparently seamless network implementations, as seen from user perspectives. Work has concentrated on the services which will be provided at ground interfaces, referred to as Service Access Points (SAPs), where services would appear externally as functions and capabilities would be made available by a providing agency to a requesting agency. (Ref. 11)

2. ARCHITECTURAL ELEMENTS

In our view, the top level architectural elements needed for an IGDS parallel those described for EOSDIS. (Ref. 12) We include a government architectural element to those to implement a global infrastructure. The top level architectural elements, then, are as shown in Figure 1.

1) User Community. This would consist of a broad range of scientific and academic users in widely dispersed facilities. Government users would access the IGDS for general policy, planning, management and operation of the IGDS, and to exert operational direction. Commercial and industrial users would include developers of applications. Individual users and organized user groups would desire access to segments of the IGDS.

2) Mission Operations. Elements which correspond to those in COSMICS and the Flight Operations Segment of EOSDIS would be needed to implement an IGDS.

3) Information Processing and Distribution ele-

ment. A system similar to the backbone network element in EOSDIS would be needed in an IGDS. We combined components of the EOSDIS Science Data Processing Segment (SDPS) and Communications and System Management Segment (CSMS).

4) Government. Governments can be expected to interface at the space agency level to provide for national goals and to establish the basis for cooperation. (Ref. 13) Organizations such as INMARSAT and INTELSAT might provide suitable models.

5) Industry. A suitable system of interfaces between industry and IGDS system management would be needed to provide for development of a wide range of services. Governments might guarantee delivery of data products to vendors at network interface points, for example.

3. IGDS FUNCTIONALITY

In order to derive communications subnetwork partitions to support a functional IGDS, we partitioned the IGDS functions into independent sets which might be accomplished in cooperation. Our analysis indicates the functional partition at Table 1 to be adequate. Our methodology paralleled Walford's architectural development framework. (Ref. 14)

We imposed the structure described by Witworth (Ref. 15) to govern the relationship between the ground systems, space segments, and data users in typical space missions. We included the major ECS functions in our analysis (Ref. 16). A discussion of each of our functional categories follows.

1) Data User Services. These functions control design and implementation of those aspects of the network which directly interact with the end user. (Ref. 17) These functions ensure that users need not know details about the IGDS to effectively use it. We formed two functional groups from these: (1) Data operations, to process and distribute bulk data, and (2) User support, to provide user interfaces.

2) Mission Operations. We based this set of functions closely on the ECS Flight Operations Segment (FOS). We formed four functional categories: (1) Aerospace craft operations performed by worldwide communications and tracking facilities, (2) Mission planning and scheduling, (3) Person-

nel training, and (4) Engineering support conducted by science investigators and national centers primarily to solve technical problems which occur during missions.

3) IGDS Management and Operations. These functional categories provide for international participation by government and industry. They extend functions assigned to international partners in the ECS.

4) Network Services. We organized these functions according to the principles explained by Walford for communications network overlays. (Ref. 18) This functional grouping reflects an overlay structure wherein the IGDS communications network is to be realized as mutually exclusive subnetworks having characteristics determined by attributes required at appropriate levels of abstraction.

Our analysis of communications network services yields the following functional categories: (1) Network Management, which includes planning, operation, and administration of the communications network, (2) and (3) The Space and Ground Segments of the communications network, (4) Process support, which provides the services and functions needed by communications networks. We included applications development functions, which consist of the tools and techniques for developing and maintaining the applications provided by the IGDS to its users, in process support. It became apparent that, from the users' perspective, data processing applications services could best be supported by the network if treated in much the same fashion as other process support services.

4. NETWORK ORGANIZATION

We partitioned the IGDS into subnetworks using the following procedure:

1) We correlated the architectural elements discussed in Part 2 of this paper with the IGDS network functions discussed in Part 3 above. Our correlation produced an assignment of each function to one or more of the elements. We distinguished between the following four levels of involvement for each assignment: (a) Primary responsibility for accomplishment, (b) Collateral responsibilities, (c) Participatory, and (d) Non-participatory.

2) Next, we grouped the functional assignments into subnetwork overlays. We prioritized each overlay with regard to the criticality of the functions it contained. Criteria included timeliness and security.

3) We then grouped the subnetwork overlays into network partitions which could be implemented and controlled independently to make the IGDS fully functional.

Our analysis indicates that communications network partitions to support an IGDS could be implemented as five independent network partitions. These are shown in Table 2 and described below.

1) An Operations high speed, highly critical top priority network, which connects the Aerospace Operations and the Information Processing and Distribution architectural elements. This network supports the core operations mission functions. Connectivity is extended to elements of the User Community element on a case-by-case basis to enable payload command functions.

2) A less critical Engineering Support network which connects the same architectural elements as the Operations network. It is an entirely off-line support environment.

3) A Data Processing network partition, confined to the Information Processing and Distribution architectural element, supports the high volume bulk data services which are at the heart of the IGDS. It connects the largest, most powerful data processors and archive facilities worldwide in support of the other IGDS functions.

4 and 5) We divided the remaining IGDS functions between two separate network partitions. Each are broadly connected to all of the IGDS architectural elements. The Communications Support network is the more critical of the two and provides the communications services. The other, the User Support and Administration network partition, provides those functions which manage the IGDS and determine the interface between the IGDS and its user community.

Our analysis indicates that an IGDS might be developed as independent but complementary implementations of the above five network partitions. As mentioned in the first section of this paper,

the CCSDS recommendations describe a wide variety of communications protocol standards which are being developed or adopted for use on space missions during the coming decades. The architecture features extensive onboard and ground computer networking within the worldwide OSI framework integrating digital transmission of video, audio, telemetry, and computer data. (Ref. 19) The CCSDS architecture supports end-to-end data communication across networks connected to the CPN via gateways using cross-support services. (Ref. 20) In the next section we review the status of efforts to develop cross-support data services.

5. DATA SERVICES

CCSDS Panel 3: System & Cross-Support has been working to develop the services needed to transfer data across networks from different agencies participating in the CPN. It has concentrated on the services which will be exchanged at the ground interface, referred to as Service Access Points (SAPs), between the networks for space operations. (Ref. 21)

The Panel defines a service as "the external appearance of functions and capabilities made available by a providing agency to a requesting agency". It developed the following four classes of 24 major services: (1) Conventional Telemetry and Telecommand, which consists of data services for these functions; (2) Advanced Orbiting Systems (AOS), which extends the packetized AOS services previously defined on the CPN across the SAPs thereby making them accessible to agencies; (3) Nontelemetric Spacecraft Monitor and Control facilities; and (4) New Services, which include facilities for reliable transport, file transfer, message service, and other services specialized to established space link services.

The scheme described by one of the Panel 3 members, A. Hooke, appears to efficiently map space link services into IGDS services in a way which can provide projects with a consistent view of services while controlling user access to the most critical of those services. Online mission control and data acquisition would, under this scheme, be provided by two independent service domains: 1) A Space Operations Data Services (SODS) segment, and 2) A Flight Operations Services (FOS) segment. SODS would provide the virtual and physical channel communications services between ground elements and aerospace craft which

require little or no mission-unique setup. FOS would provide customized services for particular projects. FOS would provide operations support and end system services, such as flight dynamics, flight monitor and control, and flight planning. A third independent service domain, the Space Applications Service Infrastructure (SPSI), would provide user data analysis and distribution services to the IGDS User Community. This structure is shown in Figure 2, which is adapted from the Panel 3 report.

6. CONCLUSION

From a functional analysis of proposed aerospace control and information systems, this paper concludes that an IGDS might be effectively supported by five independent communications networks interfaced to CCSDS cross-support service domains. Each network partition would be tailored to one of the following functional areas: (a) Operations, (b) Data Processing, (c) Engineering Support, (d) Communications Support, and (e) User Support. It proposes that the interfaces between these network partitions and aerospace craft conform to CCSDS cross-support recommendations in order to provide an integrated information and control infrastructure.

7. REFERENCES

1. Mizuta, Hiroshi et al. 1990. Information and Control Infrastructure for the Coming Space Age. In Proc 2nd AIAA/NASA International Symposium on Space Information Systems: \$3.
2. NASA Goddard RFP5-13628/307. 1991. Functional and Performance Requirements Specification for the Earth Observing System Data and Information System (EOSDIS) Core System. Goddard Space Flight Center, Greenbelt, Md.
3. (Mizuta \$1-3)
4. (NASA Goddard RFP5-13628/307 \$4.2)
5. Helgert, Hermann J., 1991. Services, Architectures, and Protocols for Space Data Systems. In Proceedings of the IEEE, 79, 9: p.1213.
6. CCSDS 701.0-B-1. 1989. Advanced Orbiting Systems, Networks, and Data Links: Architectural Specification. Blue Book, Consultative Committee for Space Data Systems: p.2-1.
7. CCSDS 100 series. 1987. Recommendations for

- Telemetry, Consultative Committee for Space Data Systems.
8. CCSDS 200 series. 1987. Recommendations for Telecommunications. Consultative Committee for Space Data Systems.
 9. CCSDS P3 Orlando Fall 1991. Report of the 9th Meeting, Panel 3: System & Cross Support, Consultative Committee for Space Data Systems, September 23 - October 2, 1991: \$2.
 10. (CCSDS 701.0-B-1 p.2-2)
 11. (CCSDS P3 Orlando Fall 1991 §4.1.2)
 12. (NASA Goddard RFP5-13628/307 §3)
 13. (NASA Goddard RFP5-13628/307 §3 and §4)
 14. Walford, Robert B. 1990. Network System Architecture. Addison Wesley: 4-30.
 15. Wertz, James R., and Larson, Wiley J. 1991. Space Mission Analysis and Design. Kluwer Academic Publishers: p.519)
 16. (NASA Goddard RFP5-13628/307 §4.2)
 17. (Walford p.19)
 18. (Walford p.18-25 and §6)
 19. Hooke, Adrian and DesJardins, Richard. 1990. Communication Protocol Standards for Space Data Systems. In Proceedings of the IEEE, 78, 7: 1295-1303 \$1-\$3.
 20. Allen, Michael A. 1990. Management and Control of CCSDS Cross-Support Services. In Proceedings of the IEEE, 78, 7: 1304-1310.
 21. (CCSDS P3 Orlando Fall 1991. §4)

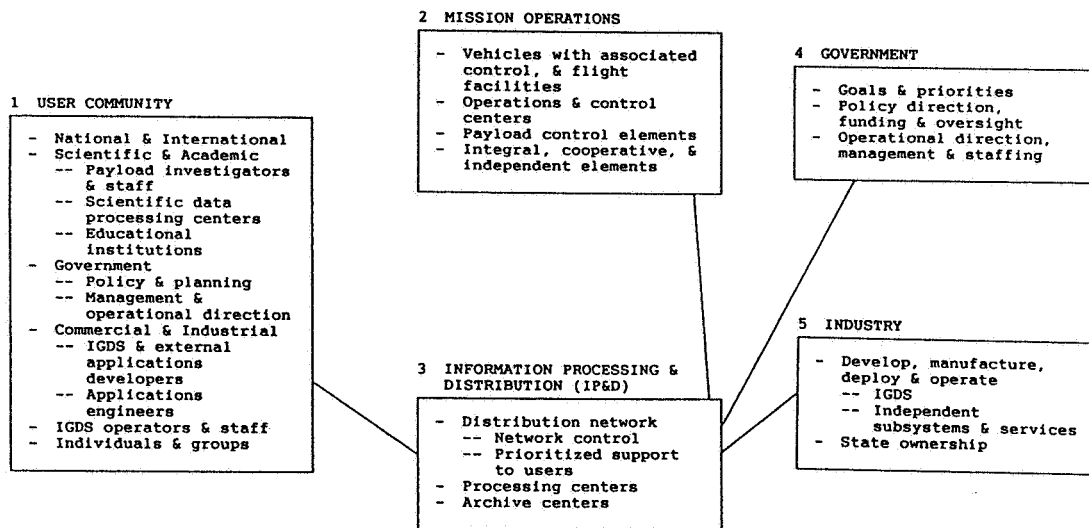


Figure 1. Architectural elements comprising an IGDS.

SERVICE	FUNCTIONS
Data User Services	1.1 Data Operations 1.1.1 Processing 1.1.2 Archive: Receive, store, and retrieve 1.1.3 Distribution 1.1.4 Auditing 1.1.5 Management: Planning, operations, & administration 1.2 User Support 1.2.1 Information and catalogs 1.2.2 Data and processing 1.2.3 Algorithm development, integration, test, and distribution
Mission Operations	2.1 Aerospace Craft Operations 2.1.1 Mission Control 2.1.2 Mission Operations 2.1.3 Payload commanding 2.1.4 Vehicle Commanding 2.2 Mission Planning
IGDS Management & Operations	3.1 Government 3.2 Industry
Network Services	4.1 Network Management 4.1.1 User Interactions 4.1.2 Maintenance 4.1.3 Network control 4.1.3.1 System performance, fault, and security management 4.1.3.2 Configuration control 4.1.3.3 Production schedule assignment 4.1.3.4 Conflict resolution 4.1.4 Accounting & billing 4.2 Space Segment 4.3 Ground Segment 4.4 Process Support: Services and functions on the network. 4.4.1 Data Processors 4.4.2 System Software 4.4.3 Communications Processors 4.4.3.1 Among Distributed Active Archive Centers (DAACs) 4.4.3.2 Interface with external networks 4.4.3.3 Network communications and processing 4.4.4 Applications Development

Table 1. Functions performed by Integrated Ground Data System (IGDS).

NETWORK PARTITION AND PRIORITY	FUNCTIONS	ARCHITECTURAL ELEMENTS
OPERATIONS PRIORITY 1	2.1.1 CONTROL 2.1.2 OPERATIONS 2.1.3 PAYLOAD COMMANDS 2.1.4 VEHICLE COMMANDS 4.1.3 NETWORK CONTROL 4.1.3.1 PERFORMANCE 4.1.3.2 CONFIGURATION 4.1.3.3 PRODUCTION SCHEDULE 4.1.3.4 CONFLICT RESOLUTION 4.2 SPACE SEGMENT	1 USER COMMUNITY 2 AEROSPACE OPERATIONS 5 INFORMATION PROCESSING & DISTRIBUTION (IP&D)
DATA PROCESSING PRIORITY 2	3.1.1 PROCESS 1.1.2 ARCHIVE 1.1.3 DISTRIBUTE 1.1.4 AUDIT 1.1.5 MANAGE	5 IP&D
ENGINEERING SUPPORT PRIORITY 2	2.4.1 ENGINEERING ANALYSIS 2.4.2 PAYLOAD HEALTH & SAFETY 2.4.3 VEHICLE HEALTH & SAFETY	1 USER COMMUNITY 2 AEROSPACE OPERATIONS 5 IP&D
COMMUNICATIONS SUPPORT PRIORITY 3	4.3 GROUND SEGMENT 4.1.2 MAINTENANCE 4.1.4 ACCOUNTING & BILLING 4.4.1 DATA PROCESSORS 4.4.2 SYSTEM SOFTWARE 4.4.3 COMMUNICATIONS PROCESSORS 4.4.3.1 ARCHIVE CENTERS 4.4.3.2 EXTERNAL NETWORK 4.4.3.3 INTERNAL NETWORK	1 USER COMMUNITY 2 AEROSPACE OPERATIONS 3 GOVERNMENT 4 INDUSTRY 5 IP&D
USER SUPPORT & ADMINISTRATION PRIORITY 4	1.2.1 INFORMATION & CATALOGS 1.2.2 DATA & PROCESSING 1.2.3 ALGORITHM DEVELOPMENT 3.1 GOVERNMENT 3.2 INDUSTRY 4.1.1 USER INTERACTIONS 2.2 PLANNING & SCHEDULING 2.3 PERSONNEL TRAINING 4.4.4 APPLICATIONS DEVELOPMENT	1 USER COMMUNITY 2 AEROSPACE OPERATIONS 3 GOVERNMENT 4 INDUSTRY 5 IP&D

Table 2. Five network partitions support the functions needed to implement the proposed Integrated Ground Data System (IGDS). Each network has the priority indicated in the first column. The remaining two columns list the functions, keyed to Table 1, performed by each network partition and the architectural elements, keyed to Figure 1, served by each partition.

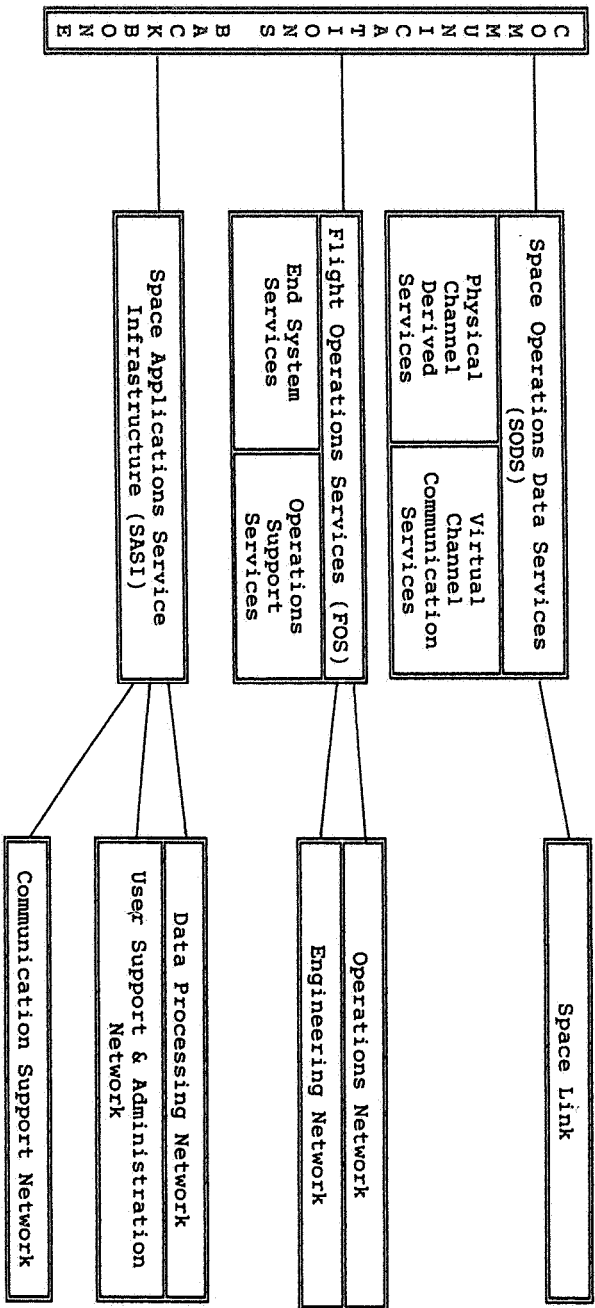


Figure 2. IGDS Infrastructure

N94-23907

THE ADVANCED ORBITING SYSTEMS TESTBED PROGRAM: RESULTS TO DATE

Penny A. Newsome and John F. Otranto

CTA INCORPORATED
Rockville, Maryland U.S.A.

ABSTRACT

The Consultative Committee for Space Data Systems Recommendations for Packet Telemetry and Advanced Orbiting Systems (AOS) propose standard solutions to data handling problems common to many types of space missions. The Recommendations address only space/ground and space/space data handling systems. Goddard Space Flight Center's AOS Testbed (AOST) Program was initiated to better understand the Recommendations and their impact on real-world systems, and to examine the extended domain of ground/ground data handling systems.

Central to the AOST Program are the development of an end-to-end Testbed and its use in a comprehensive testing program. Other Program activities include flight-qualifiable component development, supporting studies, and knowledge dissemination. The results and products of the Program will reduce the uncertainties associated with the development of operational space and ground systems that implement the Recommendations. The results presented in this paper include architectural issues, a draft proposed standardized test suite and flight-qualifiable components.

Key words: AOS Testbed, CCSDS, service and network management, SOSDU, testing, flight-qualifiable components.

1. INTRODUCTION

Goddard Space Flight Center's (GSFC's) Advanced Orbiting Systems (AOS) Testbed (AOST) Program provides the bridge between development and widespread use of the Consultative Committee for Space Data Systems (CCSDS) Recommendations for Packet Telemetry and AOS (Refs. 1-2). The Recommendations, which will be used by future NASA missions, offer the promise of significantly reducing mission life-cycle costs by standardizing data handling interfaces in both space and ground systems. Standardization allows the reuse of hardware and software, and facilitates agency interoperability in national and international programs.

The AOST Program is using a multi-faceted approach that will reduce the remaining costs and risks associated with the use of the Recommendations by

NASA. This paper presents results and products of AOST Program activities to date (November 1992). The results and products have significant implications with respect to the development of systems built in accordance with the CCSDS Packet Telemetry and AOS Recommendations.

2. AOST PROGRAM OVERVIEW

This paper is a companion paper to (Ref. 3), which provides a detailed description of the AOST Program. A high-level overview of the Program is given in this section to provide a context for the discussion of activities and results to date. Readers interested in a more comprehensive description of the AOST Program, its goals, objectives, and work activities are referred to (Ref. 3).

The AOST Program comprises five work activities: developing and using the AOST, developing flight-qualifiable components, conducting a test program, performing supporting studies, and disseminating the knowledge gained and products developed through Program activities. Detailed descriptions of these activities are presented in (Refs. 3-4).

The services defined in the Recommendations are being implemented in the AOST to demonstrate that there is at least one technically feasible, economically acceptable solution for providing each service and to perform real world validation of ground implementability in technology, operability, and manageability. *It is not intended that the AOST select a particular design solution for CCSDS services.* The AOST contains multiple implementations of many of the CCSDS services, providing a platform for testing alternative design solutions through which critical information can be provided to program management, industry, and the CCSDS.

The AOST is being developed as a series of four Capabilities, each distinguished by the CCSDS services and functions provided. The AOST comprises all of the subsystems of an end-to-end CCSDS-based data handling system, and includes both space and ground elements. Capability One (C-1) space elements include an instrument simulator and a frame generator that creates Virtual Channel Data Units (VCDUs). Ground systems include a front end, multiple service processors, and, in

Capability Two (C-2), a communications address processor and service and network management elements. The ground system elements process the VCDUs generated by the space elements to provide the CCSDS service(s) designated by the end user via the service and network management elements.

3. ACTIVITIES TO DATE

The AOST Program has produced a number of documents in support of Program activities. These documents establish Program goals and approach, allocate functions to space and ground system elements, define Testbed interfaces, and represent initial development efforts in service and network management. The AOST Program Plan (Ref. 4) defines the goals and objectives of the Program and the technical approach to achieving those goals and objectives. The Program Plan also summarizes the allocation of functions to the Testbed space and ground system elements. The Interface Definition Documents (Refs. 5-9) and the Space Operations Service Data Unit (SOSDU) Format Definition Document (Ref. 10) define the internal and external data and protocol interfaces of the Testbed. Draft specifications for service management interfaces with ground system elements (Refs. 11-12) define the management interfaces that support efficient use of resources and systems.

The C-1 AOST delivered in March 1992 provides three CCSDS AOS services: Virtual Channel Access (VCA) service, VCDU service, and Path Packet service. The AOST team established and documented ground rules governing the implementation of these services in C-1 (Ref. 13).

Functional testing of the C-1 AOST is still in progress. Research testing will be initiated upon completion of functional testing. The Master Test Plan (Ref. 14) and C-1 Test Plan (Ref. 15) define the general test program and the C-1 test program, respectively. The C-1 test program has produced a number of findings, most of which have been specific in nature and related to software debugging rather than issues related to the CCSDS Recommendations. A few significant issues have been identified and are addressed in this paper. The test program is also supporting the development of a suite of standard conformance and compatibility tests for use by CCSDS system developers.

The European Space Agency (ESA), Canadian Space Agency, National Space Development Agency of Japan, and the Deep Space Network Prototyping Laboratory at Jet Propulsion Laboratory (JPL) have expressed interest in cooperative testing using the AOST. A draft test plan for joint testing using the

AOST and ESA's Space Data Network Testbed is currently under review by NASA and ESA (Ref. 16).

The development of flight-qualifiable components is well ahead of schedule. A preliminary specification has been issued for a Reed-Solomon (R-S) encoder chip (Ref. 17).

Several members of the AOST Program team are supporting the CCSDS and the Space Operations Services Infrastructure (SOSI) working group. The SOSI group is a joint GSFC/JPL team that is developing service and operations concepts and identifying cross-support points, protocols, and formats common to all NASA implementations of CCSDS services (Ref. 18).

A library of AOST Program and related documents has been established as a central repository of knowledge gained and products of the AOST Program. The first "Workshop for CCSDS Implementors" was held at GSFC on November 4-5, 1992, under the sponsorship of the AOST Program. The Workshop was attended by more than 300 people representing 44 U.S. companies, the Department of Defense, 2 universities, 4 NASA centers, and NASA Headquarters.

4. RESULTS

The results of the AOST Program to date apply to three different aspects of the Program: the AOST architecture, a draft proposed standardized test suite, and flight-qualifiable component development. Findings related to the AOST architecture were identified through the development process and C-1 test program. The proposed test suite is a product of the C-1 test program. One of the flight-qualifiable components has been baselined by two GSFC missions.

4.1 Architecture

The architecture-related findings discussed in this section have resulted from the C-1 AOST development process and subsequent C-1 test program. Of primary importance are the ground rules identified to govern the AOST development process, the SOSDU format, service and network management, AOST scheduling and configuration, and latencies induced by the selected implementation of the AOST architecture.

4.1.1 Ground Rules

Ground rules (Ref. 13) for data processing were identified early in the C-1 development process as part of the developers' set of requirements. The domain of the CCSDS Recommendations ends at the

space/ground interface; the Recommendations do not address the requirements for implementing a ground system. The ground rules are needed to standardize ground/ground interfaces and ground processing among multiple implementations of the CCSDS services. The ground rules cover parameterization of data, processing requirements related to data characteristics, and processing requirements related to error conditions. While the AOST ground rules have not yet been proposed for adoption by the CCSDS community, it will be necessary to adopt some common set of ground rules, or for similar ground rules to be incorporated into a CCSDS Recommendation.

4.1.2 SOSDUs

The SOSDU data structure (Ref. 10) "extends" the CCSDS Recommendations to ground system interfaces. The SOSDU allows for the process-oriented decommutation of CCSDS data structures while preserving the service types and providing a mechanism for the incorporation of metadata related to the processing of the CCSDS data. The SOSDU header currently contains a generalized label and a SOSDU label common to all SOSDUs, a set of service parameters specific to each of the CCSDS services, and optional agency- and/or network-unique fields. The SOSDU is in an early stage of development and is expected to change significantly within the next several months. The SOSI group is identifying SOSDU requirements based on a global perspective of cross-support points, protocols, and formats.

4.1.3 Service & Network Management

AOST Service Management will explore ways to improve the efficiency of network operations and resource utilization in networks providing CCSDS-based services. Currently, ground systems handling space data are geared to scheduled, circuit-switched communication services. The ability to trace end-to-end data flows and determine system performance is largely dependent on verbal reporting mechanisms and operator intervention. Problems are usually detected and reported first by end users. The users must be familiar with the operation of the ground infrastructure to make use of the system's functionality.

The Service Management system being developed for the AOST (Refs. 11-12) allows users of a CCSDS service-providing network to interact with that network in terms of services rather than equipment configurations, and manages the equipment configurations of multiple facilities and subnets. AOST Service Management distributes service configuration information, generates periodic reports about the

quality of services, and monitors network elements for fault isolation. The AOST Service Management function will provide greater fault detection, isolation, and recovery capability for CCSDS data services.

AOST Service Management uses a distributed hierarchical architecture comprising a central Network Management Integrator and Coordinator (NMIC), multiple Complex Managers, and their associated managed systems. The AOST Service Management hierarchy is illustrated in Figure 1.

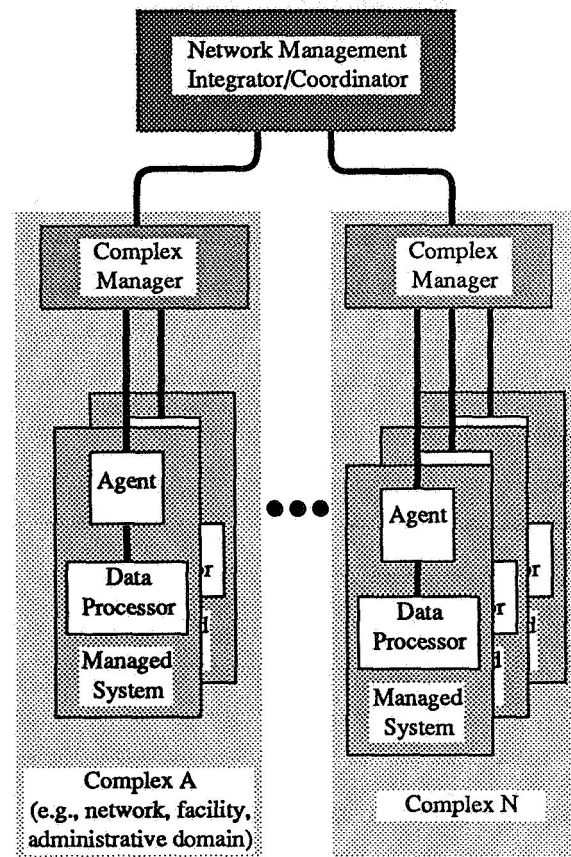


Figure 1. Service Management Hierarchy

The NMIC interfaces with the user community and coordinates the services provided by multiple complexes. Each Complex Manager manages the operation of a single complex, such as a subnetwork, facility, or administrative domain. Complex Managers manage local equipment as well as services. The systems comprising a complex must each support an agent that translates the local representation of information into a global representation and standard management protocol used by the Complex Managers and the NMIC. The AOST NMIC and Complex Managers are being implemented using SunNet Manager, a commercial off-the-shelf network management product.

Managed systems comprise two conceptual components: the data processor, which performs the actual processing and presents the agent with a local representation of managed parameters; and the agent, which translates the management information between the local representation and a global representation understood by the Complex Manager. The agent presents the Complex Manager with a view of the processor as a collection of abstract functions and system operation parameters.

The AOST is developing and testing a demonstration version of a standard Management Information Base (MIB) for CCSDS services that can be used by other CCSDS-compliant systems. The MIB includes the CCSDS AOS, Packet Telemetry, and Telecommand (Refs. 19-21) services and protocols, including configuration and performance information, and allows the NMIC, Complex Managers, and managed systems to exchange management information in a common, standard language.

4.1.4 Scheduling/Configuration of the AOST

The CCSDS Recommendations assume data-driven rather than scheduled space data systems. However, all of the C-1 AOST elements require some degree of pre-configuration prior to receiving data. Each time the data parameters change (e.g., a new Spacecraft ID or a different CCSDS service is used), each element must be reconfigured. Typically, reconfiguration is accomplished by loading a pre-defined catalog from disk to memory. If a new configuration is being used, a new catalog must be manually constructed, stored on disk and loaded into memory. This is a time-consuming process that is subject to error.

The loading of catalogs consistent with the expected data stream requires prior knowledge of the data characteristics and is a form of system scheduling. To simplify AOST operations in C-2, reconfigurations will be performed by the automated AOST service management system. The AOST will still be "scheduled," but from a centralized location.

A solution under consideration by the AOST is the use of adaptive frame synchronizers that would automatically determine the frame length of incoming data, the presence/absence of R-S coding, and that would identify the data by examining the Version ID and Spacecraft ID fields in the frame header. The result would be a truly automatic, data driven system.

4.1.5 Data Latency

Some of the elements in the AOST have been developed using a "pipeline" architecture. Incoming data are used to "push" data previously received

through the system. In the AOST environment, this pipelining occurs at the frame level, introducing unpredictable data latency. At the close of any data transmission (such as at loss-of-signal), the last several frames received will lie dormant in the system until they are manually "flushed," flushed after a system time-out, or when frame sync lock is reacquired at acquisition-of-signal. In the third case, data output from system upon reacquisition contains one or more data units that represent "old" data and some that represent "new" data, resulting in unusual and unpredictable latency.

Although unpredictable latency may not be significant to many applications, it is unsuitable for supporting real-time, interactive operations such as "joysticking." Constant latency is more important than low latency in joysticking operations because the operator needs to be able to reliably predict the response time of the system. Data latency that varies unpredictably by orders of magnitude may have serious operational consequences. Operational systems having requirements for constant or low latency must avoid a pipelining architecture.

4.2 Test Suite

An important product of the AOST Program will be the suite of tests that is being developed (Ref. 22). The test suite will be available to organizations wanting to test their existing or new CCSDS-based systems, and will serve as the basis for the conformance and compatibility test suite identified as an AOST Program objective (Ref. 4). The test suite does not currently address performance testing.

The proposed draft test suite is organized in terms of the 22 functions and 8 services defined in the CCSDS AOS Recommendation (Ref. 2). The functions can be divided into data generation functions (transmitting) and corresponding data processing functions (receiving). Specific groupings of these functions comprise procedures that in turn comprise the CCSDS services. Each function may support multiple procedures and services. Examples of functions include fill generation, VCDU commutation, and frame synchronization. Examples of procedures include Channel Access, Packet Transfer, and Bitstream. Examples of services include Virtual Channel Access and Path Packet. A complete listing of the functions, procedures, and services can be found in (Ref. 22).

The test suite provides a framework of standard tests to verify the compliance of any given implementation with the CCSDS Recommendations. Users may select from among the standard tests to accommodate the particular services and function-to-element allocations of their system implementations. Testing

occurs at the function level. After successful testing, the functions can be grouped into procedures and services which are then tested at the procedure or service level. This approach assumes that each service has been implemented using the previously tested functions as "building blocks." The alternative, in which each function is implemented separately for each service, would require independent testing for each function supporting each service.

The CCSDS Recommendations do not define or standardize the parameters or services used by each implementation. Rather, the Recommendations define only the data structures and delimiters for the services. Each flight project is free to choose which services it will support, which virtual channel IDs and application process IDs will be used, and what combinations of services and parameters will be permissible. The standardized test data suite may lead to a test data set generation capability that will permit users to select the functions and services to be tested, the particular data specifications for each implementation, and specific parameters such as Spacecraft IDs, sequence counter starts, etc. The end product of such a data set generation capability would be a test data set consistent with the CCSDS Recommendations that is tailored to the particular implementation being tested.

4.3 Flight-Qualifiable Components

Three CCSDS-based flight-qualifiable components are being developed:

- Reed-Solomon Decoder Chip Set
- Reed-Solomon Encoder
- Packetizer

The R-S Decoder Chip Set comprises four, fully custom VLSI chips using a Euclid algorithm. The Decoder Chip Set was produced by a commercial foundry and cannot be flight-qualified (Ref. 23). It corrects a maximum of 16 symbols at a sustained rate of 80 Mbps. The next-generation flight-qualifiable decoder is being developed as a single chip encoder/decoder. The new chip set will be flight-qualifiable and will perform 1 to 16 symbol error corrections at a sustained data rate of 150 Mbps.

The flight-qualifiable R-S Encoder features a selectable interleave depth (1 to 8) and supports a sustained data rate of 200 Mbps (Ref. 17). This Encoder is currently available for flight project use, and has been baselined by the Tropical Rainfall Measuring Mission and the X-ray Timing Experiment.

The Packetizer is currently being defined. The Packetizer will create CCSDS Packet headers and will support variable or fixed length packets from 64 bytes to 2K bytes. The Packetizer will provide for a 10 MHz input data rate and a 20 MHz output data rate. A preliminary specification is being drafted and will be available in December 1992.

5. SUMMARY

The AOST Program is making significant progress towards its goals of supplying information to ground and flight programs, to industry, and to the CCSDS; developing a conformance and compatibility test suite; and promoting the use of the CCSDS Recommendations for Packet Telemetry and AOS by flight projects. Architecture considerations, including ground rules, a ground/ground data transfer mechanism, service and network management, scheduling, and latency are emerging from the AOST development process and C-1 test program. Additional lessons will result from the continuation of the C-1 test program and the initiation of the C-2 AOST development process. A proposed test suite has been drafted and will be updated and revised as work on the AOST Program continues. Flight-qualifiable components are being developed and prepared for flight-qualification by flight projects.

6. ACKNOWLEDGEMENTS

The authors would like to thank Mr. Richard Carper, AOST Program Coordinator, and all of the members of the AOST Program team whose dedication to the success of the AOST Program is reflected in this paper. Thanks also to Mr. Charles Fuechsel, NASA Headquarters, for providing the funding for the AOST Program.

7. REFERENCES

1. CCSDS. 1987. *Recommendation for Space Data System Standards. Packet Telemetry*. CCSDS 102.0-B-2, CCSDS Secretariat, NASA, Washington, D.C.
2. CCSDS. 1989. *Recommendation for Space Data System Standards. Advanced Orbiting Systems, Networks and Data Links: Architectural Specification*. CCSDS 701.0-B-1, CCSDS Secretariat, NASA, Washington, D.C.
3. Carper, Richard D. 1992. NASA/Goddard Space Flight Center's Testbed for CCSDS Compatible Systems. In *Space Ops 1992 Symposium Proceedings*.

4. Carper, Richard D. January 17, 1992. *Advanced Orbiting Systems (AOS) Testbed Program Plan, Issue-1*, GSFC, Greenbelt, MD.
5. _____. March 30, 1992. *Instrument Simulator (IS)/Video Digitizer/ Packetizer/Multiplexer (VDPM) Interface Definition Document, Capability One*, CTA INCORPORATED, Rockville, MD.
6. _____. March 30, 1992. *Wideband Transfer Frame Formatter (WTFF) Interface Definition Document, Capability One*, CTA INCORPORATED, Rockville, MD.
7. _____. March 30, 1992. *Advanced Front End Processor (AFEP) Interface Definition Document, Capability One*, CTA INCORPORATED, Rockville, MD.
8. _____. March 30, 1992. *AOS Testbed Services Processor (ATSP) Interface Definition Document, Capability One*, CTA INCORPORATED, Rockville, MD.
9. _____. July 30, 1992. *Communications Address Processor (CAP) Interface Definition Document, Capability Two Version 0.1, Draft*, CTA INCORPORATED, Rockville, MD.
10. _____. February 10, 1992. *Space Operations Service Data Unit (SOSDU) Format Definition Document, Version 2.1, Draft*, CTA INCORPORATED, Rockville, MD.
11. _____. June 30, 1992. *Service Management Interface Definition Document, Capability Two, Draft*, CTA INCORPORATED, Rockville, MD.
12. _____. May 29, 1992. *Interface Specification Between the Advanced Orbiting System (AOS) Testbed Service Processor (ATSP) and the Service Management Agent for the ATSP: Capability Two, Version 1.1, Draft*, The MITRE Corporation, Greenbelt, MD.
13. _____. September 1, 1992. *Ground Rules for the AOS Testbed*, CTA INCORPORATED, Rockville, MD.
14. _____. September 30, 1992. *Advanced Orbiting Systems (AOS) Testbed Master Test Plan, Draft*, CTA INCORPORATED, Rockville, MD.
15. _____. April 30, 1992. *Advanced Orbiting Systems (AOS) Testbed Capability One Test Plan, Draft*, CTA INCORPORATED, Rockville, MD.
16. _____. September 1, 1992. *A Proposal for Compatibility Testing of NASA and ESA Testbeds, Draft*, Goddard Space Flight Center and the European Space Agency.
17. _____. _____. *Space Qualified High Speed Reed Solomon Encoder T=16, Preliminary Product Specification*, University of New Mexico, Albuquerque, NM.
18. NASA. November 1992. *NASA Concept Presentation: Space Mission Interconnection Model and Services*, presented at the CCSDS Panel 3 meeting of November 1992 at the GSFC, Greenbelt, MD.
19. CCSDS. 1987. *Recommendation for Space Data System Standards. Telecommand, Part 1: Channel Service, Architectural Specification*. CCSDS 201.0-B-1, CCSDS Secretariat, NASA, Washington, D.C.
20. CCSDS. 1987. *Recommendation for Space Data System Standards. Telecommand, Part 2: Data Routing Service, Architectural Specification*. CCSDS 202.0-B-1, CCSDS Secretariat, NASA, Washington, D.C.
21. CCSDS. 1989. *Recommendation for Space Data System Standards. Telecommand, Part 3: Data Management Service, Architectural Definition*. CCSDS 203.0-B-1, CCSDS Secretariat, NASA, Washington, D.C.
22. Parker, P.K. September 30, 1992. *Standard Test Data Suite and AOS Testbed Support for New AOS Implementations*, CTA INCORPORATED, Greenbelt, MD.
23. _____. January 1991. *Product Specification AHA4600, High Speed, Reed-Solomon, CCSDS Code, Encoder/Decoder Device Set*, Advanced Hardware Architectures, Moscow, Idaho.

N94-23908

RELIABLE TRANSFER OF DATA FROM GROUND TO SPACE

Fred Brosi

CTA INCORPORATED
Rockville, Maryland, USA

ABSTRACT

This paper describes the problems involved in uplink of data from control centers on the ground to spacecraft, and explores the solutions to those problems, past, present, and future. The evolution of this process, from simple commanding to transfer of large volumes of data and commands is traced. The need for reliable end-to-end protocols for commanding and file transfer is demonstrated, and the shortcomings of both existing telecommand protocols and commercial products to meet this need are discussed. Recent developments in commercial protocols that may be adaptable to the mentioned operations environment are surveyed, and current efforts to develop a suite of protocols for reliable transfer in this environment are presented.

Key Words: Space communications, CCSDS, telecommand, ARQ

1. INTRODUCTION

Transfer of data from control centers on the ground to spacecraft has evolved from sending a few discrete commands, to uploading of command sequences, software modules, and data files. Data rates and volumes have increased by several orders of magnitude over the past 35 years. Total round-trip delay times have increased, due to increased processing and use of LANs both on board and on the ground, and to use of geosynchronous data relay satellites.

In parallel, increasingly sophisticated techniques have been developed to move data from ground to space. Early manual techniques have been replaced by Automatic Retransmission Queueing (ARQ) protocols designed to work in the space link environment. These telecommand protocols meet the basic requirements of complete, in-sequence delivery, but they operate only across the space link, and thus cannot provide reliable transfer end-to-end. They provide only primitive file transfer capabilities, and some require careful monitoring by operators.

Commercial protocols provide many of the end-to-end capabilities needed for space operations, but they were developed for ground-based, multi-user networks, and implementations of these protocols are often optimized for that environment. These protocols lack some of the protocol features needed for efficient use of the scarce resources of the space link (Ref. 1).

The paper concludes by noting recent developments in commercial protocols that may be adaptable to the space operations environment, and the plans of the Consultative Committee for Space Data Systems (CCSDS) to develop Recommendations to meet the space data transport needs of future space missions.

2. EVOLUTION OF COMMANDING

2.1. No Uplink

The first few spacecraft launched in the 1950s had no ability to accept and act on commands from the ground. They simply switched on as they went

into orbit, and transmitted their telemetry signals continuously. Mission objectives were simple, and there was little to control. But before long, mission objectives, and the instruments and spacecraft systems that were sent into space to meet these objectives, became more complicated. We needed to be able to operate the satellite from the ground by sending commands.

2.2. Send & Hope

Early command requirements were modest by today's standards, often limited to turning a few pieces of equipment on or off, or starting or stopping an on-board operation. Sometimes an operating mode was changed. All of these simple operations could be accomplished by use of single, discrete commands. Since each command produced an effect on the spacecraft, the operator could verify that a command had reached the spacecraft and been correctly executed by watching for the desired effect in telemetry on the downlink. The spacecraft had no ability to evaluate the correctness of the sequencing of commands.

Commands sent to a spacecraft do not always arrive intact. Transmission noise, signal fading, or interference could lead to loss of commands. Thus if several commands had to be performed in sequence, the operator would have to verify each command before sending the next one. The communications function of checking for delivery of the commands was performed by the operator and the telemetry system rather than by a communications protocol.

2.3. Send & Count

Later, command procedures were based on a spacecraft command counter, which was incremented each time a new command arrived. The spacecraft also had the ability to reject the

remainder of a sequence of commands after detecting an error. The value of the command counter, sent in telemetry, allowed the operator to verify receipt of an entire sequence, or to retransmit any rejected commands. A Barker code at the beginning of each command sequence would unlock the spacecraft, clearing any previous error conditions.

A further refinement of this approach uses sequence numbers in the data units that carry commands. These numbers can be checked on the spacecraft to assure that commands arrive in sequence. This approach is used in the CCSDS COP-1 telecommand Protocol (ref. 2).

2.7. Procedure vs Protocol

As the approaches to commanding described above show, spacecraft telecommand involves a combination of *protocol*—the communications technique used to transfer commands and to control and verify that process, and *procedure*—the actions that the operator must perform to use the protocol and other ground facilities.

Since there are strong drivers to keep space hardware and software simple, commanding protocols have historically been minimal, with the burden falling on operations procedures. But minimal protocols also limit throughput, and as the volume of command and data to be sent from Ground to space increases, more complex protocols are needed to provide this transfer efficiently and reliably.

3. FUNDAMENTAL REQUIREMENTS

The fundamental requirement for a telecommand protocol is complete transfer of a sequence of command data units (CDUs) to the spacecraft in order, and without duplication.

3.1. Completeness

Completeness requires that all CDUs sent from the ground must arrive on board. Either the telecommand protocol or the operations procedure for its use must assure that no CDU goes undelivered. Special features must sometimes be included in the protocol to ensure that the first and last CDU in a sequence is received correctly.

For example, using COP-1, the spacecraft checks sequence numbers to determine if a CDU is missing, but if the last CDU is lost, the spacecraft sees no gap in sequence numbers, and so cannot detect the failure. A timer in the COP-1 sending process is used to initiate retransmission if the last CDU of a sequence goes unacknowledged for too long.

3.2. In Order

In-order delivery requires that if a CDU is lost (e.g., due to an uncorrectable error), any subsequent CDUs in the sequence must not be delivered to their destination on the spacecraft until the missing CDU has been retransmitted and correctly received.

3.3. No Duplication

The requirement for no duplication has a significant impact on the strategy used for retransmitting CDUs when transfer is interrupted. Either the ground must be prevented from sending a duplicate of a command that has already been received by the spacecraft, or the spacecraft must be able to detect and reject duplicates.

4. EVOLVING OPERATIONAL REQUIREMENTS

4.1. Telecommand Context—The CPN

The CCSDS Recommendation for Advanced Orbiting Systems ((ref. 3) defined the concept of the CCSDS

Principal Network (CPN), which is the concatenation of on-board, space link, and ground subnetworks. The CPN is the context within which a telecommand protocol operates. As such a protocol is developed, design decisions often depend on the locations of the points within the CPN between which the protocol will operate.

The increasing use of on-board LANs raises the probability of data loss between on-board receipt over the space link and final delivery at the on-board destination. This makes a transport level protocol desirable as a replacement for the link level telecommand protocols now in use.

4.2. High volume Uplink

Larger on-board memories and the need to reduce operations costs through autonomous spacecraft operation is increasing the volume of data transferred from space to ground. This data includes stored command loads, as well as data base updates and software loads. This increase in traffic requires higher rates and higher throughput from telecommand protocols.

4.4. Delay Time

Use of data relay satellites, on-board networks, and more complex ground networks, leads to longer delays. Some multi-spacecraft missions need to pass messages through space-space links to reach end point. These delays, which are orders of magnitude greater than typically seen in ground networks, affect the selection of protocol functions and sizing of protocol parameters. Deep space missions have extremely long delays, and so present a special challenge to designers of space transport protocols.

4.5. Multi-Session Uploads

The need to transfer large data files during short contacts with earth satellites, or over low data rate links to deep space probes leads to a need to spread an upload operation over two or more contacts. Thus a primitive session layer protocol is needed to allow suspension of a transfer operation and restart from a checkpoint.

4.6. Operations Costs

Operating costs are a significant part of the total cost of space missions, and sometimes are the determining factor in limiting the lifetime of a mission. Increasing the level of automation in telecommand protocols simplifies operational procedures, and so decreases the need for manual monitoring and control. As a side benefit, taking the man out of the loop can reduce delay, thus increasing the productivity of a short contact.

5. CURRENT SOLUTIONS FALL SHORT

5.1 COP-1

The current CCSDS Telecommand protocol, COP-1 (Ref. 2), provides all the features needed for reliable transfer at moderate data rates across the Space Link Subnet, but does not provide verification of end to end transfer, and does not support high rate, long delay, or multiple contact transfers.

5.3. SLAP

The Space Link ARQ Procedure (SLAP) defined in (Ref. 3) can transfer data at higher rates and longer delays than COP-1, but it also operates at the link layer, and so is unsuitable for spacecraft with complex on-board networks.

5.2. Send & Dump

The send, dump, compare, and retransmit method used for memory

loads in the past has many drawbacks. This method wastes downlink bandwidth and contact time. If an errors are found, either the entire load must be retransmitted, with the same probability of error as in the original transmission, or a new load must be prepared to replace the portions containing the errors. The latter approach is more efficient, but often must be customized to the particular spacecraft, and so is not an approach that promotes standardization.

6. COMMERCIAL PROTOCOLS ARE NOT THE ANSWER

Although the functions needed to support ground to space data transfer are similar to those performed by protocols that operate in ground networks, these commercial, off-the-shelf (COTS) protocols have been designed to work in a different environment from that of spacecraft operations, and efforts to use commercial implementations have, to date, been disappointing. There are two classes of problems: protocol design problems, and protocol implementation problems.

6.1. Protocol Design Problems

6.1.1. Excess Baggage

Most of the protocols that are candidates for space transport protocols have many features and options that are unnecessary or even undesirable. These may include multipoint addressing, extended connection setup procedures, and other features needed in open systems, but which add to the overhead of the protocol data units.

6.1.2. Missing Features

Existing transport protocols, such as TCP (Ref. 4) and TP4 (Ref. 5), lack two important features needed in an efficient spacecraft transport protocol. The first is a means for the receiving

end on the spacecraft to indicate that a retransmission is necessary. Both TCP and TP4 provide for acknowledgement of correctly received data units, but retransmit only when a timer at the sending end expires after a data unit has gone unacknowledged for too long. Given the long, and sometimes variable, delays that are encountered in space operations, this timer must be set to a fairly high value to avoid unnecessary retransmissions. This leads to long periods of unproductive use of the uplink, during which new data units are being sent, only to be rejected by the spacecraft because of an earlier error. This can be avoided by use of an explicit request for retransmission upon detection of an error. Such a feature is provided in the COP-1 and SLAP protocols, and could easily be included in a transport protocol. In (Ref. 6), Zhang concludes that "...we should use external events as a first line of defense against failures, and depend on timers only in cases where external notification has failed."

The second missing feature is an effective means of flow control. Most ARQ protocols use a "credit" or "window" method of flow control, in which the receiving end grants the sending end credit to send a given number of frames before getting an acknowledgement. This works well in the short delay environments in which these protocols were designed to operate. But in the space operations environment, this method can lead to stuttering, in which periods of rapid transmission alternate with idle periods, and under the worst circumstances, result in severely reduced throughput. The causes of this behavior are described in (Ref. 7), along with recommended mechanisms for overcoming it. One mechanism is a rate parameter, which allows the receiving end to regulate the release of CDUs from the sending end to avoid buffer overflow. The SLAP provides this mechanism.

6.2. Implementation Problems

The second class of problems with COTS protocols involves the implementation of the protocols as commercial products. Although these products may provide excellent performance in the ground network environment for which they were designed, experiments show that throughput under nominal space operations conditions can be as low as 5% (Ref. 8).

In (Ref. 8), the factors that contribute to poor performance of a TP4 product under typical space operations conditions are identified. The situation is complex, but is dominated by design decisions that avoid retransmission, under the assumption that data transfer over the connection is to be minimized. When retransmission is needed, the software allows the link to remain idle while waiting to see if a single retransmitted data unit is acknowledged, before trying to send another. In ground networks, where the links are shared by other users, this method controls cost for the user of the protocol, and frees link capacity for other users. But in space operations, the connection cost is dominated by the scarce contact time, and there are often no other users, so this method is counterproductive.

It may be possible to adapt a commercial product to meet space transport protocol requirements, but surgically removing the parts that are not needed may be a more difficult job than building a lean, focused protocol from scratch. Even if the job of extracting the needed subset could be accomplished at a reasonable cost, the interoperability of implementations derived from two different products would be questionable. The cost of space qualifying such software would be at least as great as for an implementation of a customized space transport protocol.

7. FUTURE EFFORTS

Despite the drawbacks to using COTS protocols directly in space operations, these protocols provide a sound starting point for design of leaner, customized space transport protocols. The protocol specifications provide a menu of potential techniques, as well as an established vocabulary for the designers. COTS products can be used for prototyping and testbed evaluation, which helps to identify protocol features that must be modified to achieve needed performance in space operations.

Efforts to develop protocols for very high speed, moderate delay ground networks (Ref. 7) may produce protocols which can be adapted for use in space operations.

The CCSDS has recently begun work to define reliable protocols to meet the evolving needs for end to end transport and file transfer in space operations. This effort will begin with the definition of the functional, performance, and operational requirements for Space Transport and Space File Transfer Protocols.

8. REFERENCES

1. Chitre, Dattakumar M., and Lee, Hsi-Ling 1990. Operation of Higher Layer Data Communication Protocols over Satellite Links. In *Proceedings of the IEEE* Vol. 78, No. 7, July 1990.
- 2 CCSDS. 1992. *Space Data System Standards. Telecommand, Part 2 - Data Routing Service: Architectural Specification. CCSDS 202.0-B-2*, CCSDS Secretariat, NASA, Washington, DC.
- 3 CCSDS. 1989. *Recommendation for Space Data System Standards. Advanced Orbiting Systems, Network and Data Links: Architectural Specification.*
4. Postel, J. 1981. *DoD Standard Transmission Protocol*. ARPA RFC-793. Sept. 1981.
5. ISO/IEC. 1986. *Information Processing Systems - Open Systems Interconnection - Transport Protocol Specification*. International Standard 8073.
6. Zhang, Lixia, 1986. Why TCP Timers Don't Work Well. In *Proc. ACM SIGCOMM Symposium on Communications, Architectures, and Protocols*, Stowe, VT, 1986
7. La Porta, Thomas F., and Schwartz, Mischa 1991. Architectures, Features, and Implementation of High-Speed Transport Protocols. In *IEEE Network Magazine*, May 1991.
8. Durst, Robert C., Evans, Eric L., and Mitchell, Randy C. 1991. The Effect of Long Delay and Transmission Errors on the Performance of TP-4 Implementations. In *Proc. IEEE TriComm 91*.

N94-23909

THE EURECA TELECOMMANDING CHAIN: EXPERIENCE WITH PACKET TELECOMMAND AND TELEMETRY SYSTEMS.

C. Müller, R. Bater

Logica Space and Communications Ltd., London, UK

E. M. Sørensen

European Space Operations Centre, Darmstadt, Germany

ABSTRACT

The European Retrieval Carrier (EURECA) was launched on its first flight on the 31st July 1992 by the Space Shuttle Atlantis. EURECA is characterised by several new on-board features, most notable Packet Telemetry and a partial implementation of Packet Telecommanding using an early version of the Command Operation Procedure (COP-1) protocol. EURECA has also very low contact time with its Ground Station, with a consequent high number of out-of-visibility on-board operations. This paper concentrates on the implementation and operational experience with the COP-1 Protocol and the effect the short ground contact time has on the design of the Commanding System. Another interesting feature is that the COP-1 is implemented at the Control Centre rather than at the Ground Station. The COP-1 protocol also successfully supported the mission during the Launch where commands were sent via NASCOM and the Shuttle.

Key Words: Packet Telecommanding, COP-1 protocol, Command Verification.

1. INTRODUCTION

The European Retrieval Carrier (EURECA) is a reusable platform supplying power, cooling, ground communications and data processing services to a variety of independently-operated payloads (ref 1). Fifteen experimental facilities are carried to support more than fifty individual experiments, most relying on the microgravitational environment. The operational altitude is 500 Km. The Operations Control Centre (OCC) is at ESA's European Space Operations Centre (ESOC) in Darmstadt, Germany. The primary groundstations are at Maspalomas in the Canary Islands and Kourou at French Guinea. During the deployment and retrieval phases contact is maintained via the NASA Communications Network and the STS.

At ESOC, operational data processing is carried out on the Eureka Dedicated Computer System (EDCS) that hosts the mission-configured Spacecraft Control and Operations System (SCOS) (ref 2) and the Eureka-Specific Software (ESS) applications.

The Eureka-A1 mission has characteristics differing quite considerably from those of missions hitherto supported at ESOC. One of these is the use of Packet Telemetry and Packet Commanding. Eureka is the first ESA application of Packet Telemetry and Commanding.

2. PACKET TELEMETRY AND COMMANDING

2.1 TELEMETRY

Eureka's telemetry is packetised according to standards based on a Recommendation of the Consultative Committee for Space Data Systems (CCSDS), ref. 3.

The Packet telemetry recommendation (ref. 3) uses two principal data structures, the source packet and the Transfer Frame, source packets being multiplexed within transfer frames. Each on-board source must label its data packets using CCSDS defined headers. The transfer frames are of fixed length, optimised for high-performance transfer to the ground. For Telecommand verification each Transfer Frame has attached a Command Link Control Word (CLCW) that is used by the COP-1 Protocol. The CLCW is essentially meant to be used in automated transmission/retransmission processes. Therefore, CLCW data must be delivered error free to the sending end of the Telecommanding system. The protocol information contained in the CLCW is such that it is not required that the Telemetry Transfer Frame rate match the Telecommand Block Rate. However, some minimum CLCW sampling rate must be established for the proper operation of the COP-1 protocol.

2.2 TELECOMMAND

Unlike the telemetry system, the Eureka telecommanding system has to support the old command standards (Ref. 4) and the new Packet command standard (Ref. 5). The reason for this is the way it has been implemented on board. Command decoders using the old standard have been used as a basis, but the extra services of the packet commanding have been built into the On-Board Computer (OBC). Thus when the OBC is nominally activated, the commanding system acts like a packet command system, using a subset of COP-1 of the standard (ref. 5). If the OBC is off, the old standard has to be used.

3. COP-1 PROTOCOL

NOTE: In this section, although the word COP-1 is used, EURECA has only implemented a subset of the COP-1. The EURECA terminology and services are not completely compatible with the latest issue of the CCSDS recommendation.

COP-1 is a closed-loop Telecommand Protocol that utilises sequential ("go-back-n") retransmission techniques to correct Telecommand Blocks that were rejected by the spacecraft because of error. COP-1 allows Telecommand Blocks to be accepted by the spacecraft only if they are received in strict sequential order. This is controlled by the necessary presence of a standard return data report in the telemetry downlink, the Command Link Control Word (CLCW). A timer is used to cause retransmission of a Telecommand Block if the expected response is not received, with a limit on the number of automatic retransmissions allowed before the higher layer is notified that there are problems in sending Telecommand Blocks. The retransmission mechanism ensures that:

- No Telecommand Block is lost
- No Telecommand Block is duplicated
- No Telecommand Block is delivered out of sequence

The COP-1 protocol has also an expedited service. This service is used for exceptional spacecraft communications. Typically, this service is required for recovery in absence of telemetry downlink (i.e. no CLCW), or during unexpected situations requiring unimpaird access to the spacecraft data management system.

Figure 3.1 gives a simplified overview of the EURECA COP-1 Protocol.

COP-1 Protocol

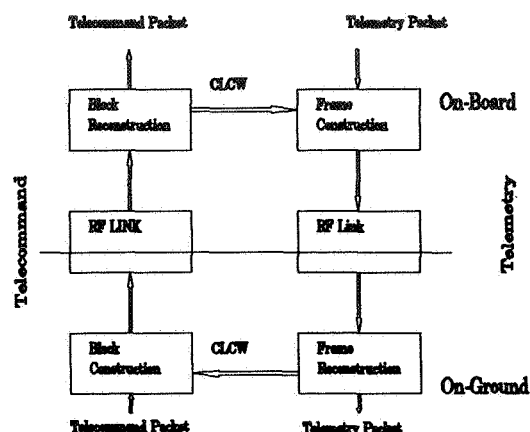


Figure 3.1 EURECA COP-1 Protocol

4. IMPLEMENTATION OF THE COP-1 PROTOCOL

The EURECA Commanding system only uses a small subset of the COP-1 services and the commanding system was designed before the COP-1 standard was available. In the event, it proved impossible to develop this general infrastructure equipment containing the EURECA standard and the COP-1 in the required time scale. It was therefore decided to use a Mark II Telecommand decoder (supporting the old ESA standard) and implement all the necessary packet block building and protocol execution in software within the spacecraft control system at the Control Centre.

Figure 4.1 gives an overview of the EDCS implementation of the COP-1 Protocol when commanding via an ESA Ground Station.

The basic principle is that all the additional packet block building and protocol execution associated with the Packet Telecommand Standard are performed within the spacecraft control system at the OCC. The ground network and station equipment is only used to transport these data structures from the control centre to the spacecraft.

4.1 COP-1 VIA NASCOM

For Eureka the possibilities exist to transmit telecommands via NASA facilities (NASCOM/MCC) and via ESA ground stations. The first is only needed during the launch and early orbit phase and retrieval. The latter is in addition needed for the routine phase. The commanding via

COP-1 IMPLEMENTATION

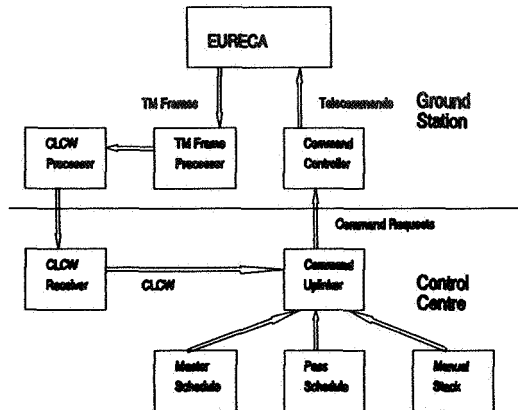


Figure 4.1 EURECA Implementation of the COP-1 Protocol

NASA facilities is complex, for two reasons: first the different NASA ground system for commanding. Secondly, the users required a command interface that looked the same as that for commanding via ESA stations (except possibly for response times); they require for example a similar manual command interface and automatic commanding; where real-time telemetry is available on-line they also require common verification.

Figure 4.2 gives an overview of the EDCS implementation of the COP-1 protocol when commanding via NASA.

COP-1 IMPLEMENTATION (VIA NASA)

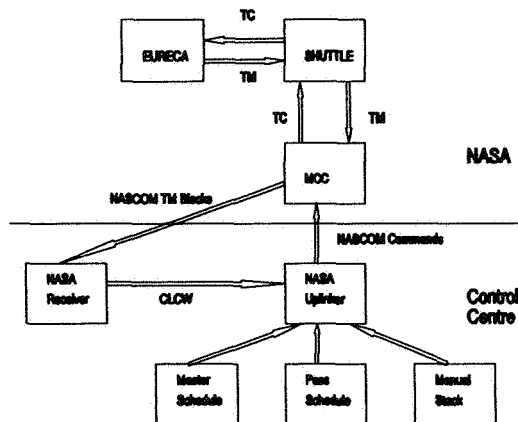


Figure 4.2 EURECA implementation of COP-1 via NASA

The same basic principles apply as commanding via an ESA ground station. The NASA infrastructure is only used to transport EURECA data structures from the control centre to the spacecraft.

4.2 CONSIDERATIONS FOR THE IMPLEMENTATION OF THE COP-1 PROTOCOL

The COP-1 protocol is only designed to provide the services between two COP-1 protocol machines one on-ground and one on-board. In terms of network layers the COP-1 protocol can be classified as a transport layer. The COP-1 does not provide an end-to-end transport service. It is the responsibility of the implementer to develop any required Higher Layer protocols using the underlying services of the COP-1 protocol. Additional protocols may also be required when multiple sources on the ground are accessing the same COP-1 machine.

The COP-1 Sequence-Controlled Service is normally initiated by means of Service Directives. However in case of ground failures it may be necessary to include additional higher layer protocol elements to initiate the COP-1 services properly. This is also required to resynchronise between multiple ground users and on-board users. To allow EURECA to operate autonomously, EURECA executes commands from its on-board Master Schedule. To support the uplink of the Master Schedule EURECA has implemented a command insert counter which is reported in the Housekeeping Telemetry. This counter is used in operational procedures to restart the uplink of the Master Schedule in case of any ground failure.

4.3 OPERATIONAL EXPERIENCE WITH COP-1

There have been a number of occasions where the COP-1 protocol has successfully recovered an error. These cases all concern link degradation, and involved the following circumstances:

1. During the Deployment phase with a bad RF link between the Shuttle and EURECA
2. During the Deployment phase where the EDCS did not receive a Command Acceptance Pattern (CAP) from NASA.
3. During ESA ground station passes where the spacecraft was configured with the wrong antennae.
4. During ESA ground passes where commanding was executed down to 0°

elevation (resulting in degradation of the telecommand and telemetry links).

5. During on-board antenna switch over.
6. When the OBC failed to allocate a Telecommand buffer (due to an OBC overload condition).

Although not all of the above cases were foreseen in the design of the COP-1 protocol (in particular case 2 and 6) the COP-1 protocol has always successfully recovered the error with a maximum of two retries. It is also important that during EURECA routine operations with a normal link budget the COP-1 protocol has never been in retry (i.e no transmission errors).

5. LOW CONTACT RATIO

5.1 MAIN DRIVER REQUIREMENTS ON COMMANDING SYSTEM

The majority of EURECA's operations are executed outside contact periods. Thus a command uplinked at time T may be time-tagged for execution at T + 10 Hours and so will be held on the on-board master schedule. The relevant necessary telemetry to verify the execution of this command might not be downlinked until T + 18 Hours, and subsequently processed to yield the final verification result at T + 19 Hours. This is illustrated schematic in figure 5.1.

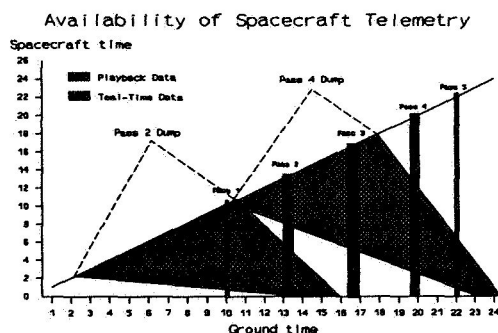


Figure 5.1 Relation between spacecraft generated data and the arrival at the OCC.

Figure 5.1 shows an example of five passes starting with a short pass followed by three long passes and ending with a short pass. During this pass sequence the on-board mass memory unit is dumped twice

(pass 2 and pass 4). The first pass 2 dump telemetry is started to be routed to the OCC shortly after the end of pass 2 and terminates before pass 3. The routing of pass 4 dump telemetry is as shown delayed until after the last pass 5.

This delay result in three significant requirements on the ground system.

1. The Master Schedule's acceptance of time-tagged commands and their subsequent release to end users must be emulated. This on-ground maintained image of the on-board commanding provides the operations staff an indirect presentation of the on-board commanding activities and the status of verification.
2. Before each pass the commanding system shall prepare, an 'expected status of the spacecraft' using a defined subset of housekeeping parameters. This is prepared under the assumption that all commands not known to have failed have succeeded. As the telemetry parameters are received they are compared with the expected values, alarms being raised when differences are detected. This 'quick look' approach may detect command failures hours before traditional verification can take place and permit corrective actions to be taken within the same pass.
3. The commanding system must be able to verify commands using Real-Time Telemetry and Playback Telemetry. For EURECA playback telemetry will never arrive interleaved with real-time telemetry. However playback must arrive in chronological order, but the time of arrival is not significant.

5.2 COMMAND VERIFICATION

The basic mechanism for command verification is that all commands to be verified will have a verification window. The start of this window is the earliest time that the telemetry could be affected by the execution of the command and the end of the window is the time at which, if the command executes successfully, the telemetry must have been affected. If the verification criteria are met in the telemetry during the time window then the command is passed successfully; if the conditions have not been met at the end of the window then the next telemetry received decides pass or fail of the command.

For EURECA the Command Verification is a four stage process consisting of:

1. Uplink verification using the Command Link Control Word (CLCW). This cannot be applied for commands executed from the master schedule.
2. Verification of delivery at on-board destination using the end user generated Acknowledgement packets.
3. Execution Verification using special EURECA Report Packets.
4. Execution Verification using the Housekeeping Packets.

Figure 5.2 illustrates the EURECA verification concept.

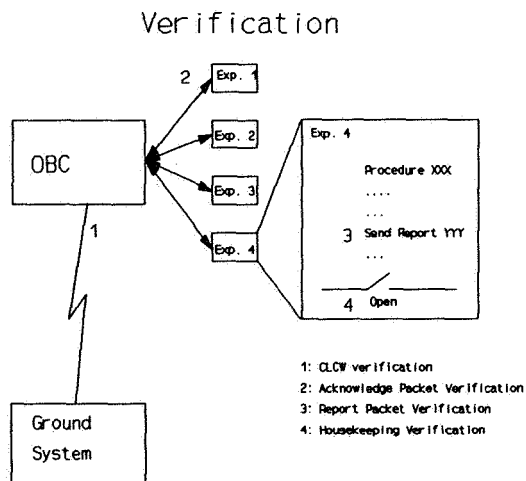


Figure 5.2 Verification

Command verification is therefore described by a vector, rather than a simple success/fail indication. Thus an example; could be 'uplink successful', 'acknowledged in real time', 'report packet received', 'housekeeping confirmation not yet available'. Some 26 such combinations are possible. Command verification also involves processing of various special packets associated with commanding, namely Acknowledge, Report and Exception packets, as well as traditional verification based upon changes in telemetry parameters. These may be found in real-time Telemetry and/or Playback Telemetry, which arrives at the OCC at different times.

6. LESSONS LEARNED

The following lessons have been learned about packet telemetry and telecommand systems from development of the Eureca spacecraft control system and operations:

6.1 COP-1 PROTOCOL

1. The COP-1 protocol has proven to be very reliable and is able to recover transmission error with minimal operational impact.
2. It is possible to implement the COP-1 protocol at the Control Centre using existing infrastructure such as the ESA MARK II Telecommand Controller and the NASCOM Remote Command Facilities
3. The design of the Control Centre must consider end-to-end protocols and provide elements that make it possible to recover in case of ground failures. The design must also consider the infrastructure to be used.
4. Introduction of Packet Telemetry and Telecommand is a major step towards standardisation of on-board and ground systems. To fully archive this goal it will be necessary to define standards governing end-to-end protocols such as File Transfer Protocol that are required to maintain on-board Master Schedules, and to support on-board software maintenance.
5. The distribution of protocol handling between the Ground Station and the Control Centre has to be considered taking into account the mission requirements. In the EURECA case where commanding is via ESA ground stations and via NASA it is desirable to implement the protocol at the control centre. This implementation also allows the COP-1 to recover in case of ground link errors. In other cases it might be considered to close the COP-1 protocol at the Ground Station. This might be attractive if commanding is considered time critical because all time critical functions are concentrated in the front-end Ground Stations. This is particularly true for the closed-loop operations of the COP-1 protocol.

6. It is not recommended to make the uplink protocol transparent to the operational staff. They require full visibility of the status of the uplink process including the status of the retry process. This is important because considerable uplink delay may be introduced when COP-1 protocol performs retries.

6.2 LOW CONTACT RATIO

1. The expected status is proven to be very useful. It gives the operational staff a prewarning and often triggers an emergency dump of on-board stored telemetry necessary for further investigation of the problem.
2. The EURECA verification implementation is based on one verification window per command. However this window can be closed by the arrival of any Telemetry Packet. This has caused problems because some on-board users have problems maintaining time synchronisation with the OBC and their local clock have therefore drifted into the future. The arrival of a packet with future time causes verification windows for other commands be falsely closed. It should therefore be considered to make the closing of the command verification window telemetry packet specific.

7. CONCLUSION

At the time of writing (October 1992) EURECA has been successfully supported by the Control Centre for 3 months. During this time the EDCS has successfully received and processed over 10 million telemetry packets and over 60000 commands has been sent. The COP-1 protocol has proven to be very reliable and given the operational staff much confidence in the EURECA commanding system.

The low contact ratio for EURECA has proven not to be a major problem. The functions provided by the EDCS gives the operational staff sufficient information for quick assessment of the status of the spacecraft during the short passes.

8. REFERENCES

1. Andresen, R.D. and Nellessen, W. "Eureca in the Columbus Scenario," 1988, Space Technology 8, 45
2. Kaufeler J-F and Mazza C., "A New Generation of Spacecraft Control System - SCOS," 1988, ESA Bulletin 56.
3. Consultative Committee for Space Data Systems, 1984, 'Recommendation for Space Data System Standards: Packet Telemetry', "Blue Book".
4. PCM Telecommand Standard ESA PSS-45 Issue 1 April 1978
5. Consultative Committee for Space Data Systems, 1986, 'Recommendation for Space Data System Standards: Telecommand, Part-2: Data Routing Service, Architectural Specification', "Blue Book".

VERIFICATION OF TECHNICAL ELEMENTS OF THE ADVANCED SPACECRAFT BASED UPON THE CCSDS RECOMMENDATION

N 9 4 - 2 3 9 1 0

*¹Hideo Hara, *¹Sigeo Yamada, *¹Yoshikatsu Yamamoto

*²Seiichi Fujii, *²Kaoru Tumura, *²Masato Yokomizo, *²Hiroyuki Ogasawara

*¹National Space Development Agency of Japan
Tracking & Data Acquisition Department
Dai-ichi Daimon Bldg. 2-10-1 Shiba-Daimon,
Minato-ku, Tokyo 105, Japan

*²Fujitsu Limited
Systems Section, Space Systems Dept.
Systems Engineering Group
1-17-25, Shinkamata, Ota-ku, Tokyo 144, Japan

ABSTRACT

We are going to meet the era when advanced spacecrafts such as the space stations are developed and operated. In the era, the current system of the satellite operations control will change largely. We consider that the future system is required to have the function as follows: the function for interchanging data between international agencies, processing the various kind of space data, and distributing data as a lot of unspecified users require.

However, We have now the following problems in order to satisfy these requirements: the problems for standardization of space data communication protocol, establishment of multimedia data management method, and standardization of user interface. This paper describes three technique to solve the above-mentioned problem. That is,

- a. Standardization of the data communication protocol between space and ground by AOS(Advanced Orbiting System) protocol of CCSDS(Consultative Committee for Space Data Systems) Recommendation,
- b. Management of multimedia data by catalog reference,
- c. Standardization of user interface by SFDU(Standard Formatted Data Unit) of CCSDS Recommendation.

Key Words: CCSDS, AOS, SFDU, multimedia, spacecraft

1. INTRODUCTION

NASDA(Nation Space Development Agency of Japan) schedules the launch of space station JEM(Japan Experiment Module) in the summer of 1998. Now, we can say the subject for space development is changing from current spacecraft such as satellite to advanced spacecraft such as space station and platform type spacecraft. The advanced spacecraft is different from the current spacecraft by the following characteristic.

At First, advanced spacecraft program is the international program; it is necessary to cooperate internationally because the development size and cost of it becomes much larger. The next, a lot of payload is frequently exchanged on advanced spacecraft for implementing various experiments. The number of payload users is increasing according to the variety. The last, multimedia data occurs such as experiment data, video data monitoring the experiment, voice data, and text data including mission plan etc. besides telemetry data. As a result, it is thought that the future system for the advanced spacecraft operations control is changing.

We have done the following work in order to acquire technical elements necessary to develop the future system.

1. Analyzed the technical problems of the future system
 2. Developed the prototype system introducing some technique
 3. Evaluated the availability of the technique
- This paper intends to describe technical elements we have acquired through these works.

2. ANALYSIS

In this section, we are going to analyze technical problems of the future system from the viewpoint of the system development.

2.1 Requirement

To sum up the characteristic of advanced spacecraft (described in section 1.), the following requirement is thought to be more important in the future system:

- Interoperability
- Multimedia data management
- User interface

That is to say, the current system has data communication protocol conforming to the standard of Japan and/or NASDA, mainly processes the digital data such as telemetry, and distributes the experiment data to a few of specified users within NASDA.

On the other hand, the future system is required to interchange space data between international agencies, process the various kinds of space data such as video and voice, and distribute data as a lot of unspecified users require.

2.2 Problem

As a result, we have found that it is necessary to solve the following problems in order to satisfy these requirements.

The first, it is necessary to determine which protocol we use, and verify the availability of the protocol. The next, it is necessary to determine how we synchronize one data with another, because space data such as telemetry, image, orbital element is relevant to each other. The last, it is necessary to determine by which method we exchange data with users, and verify the availability of the method.

To sum up, the following are problems:

- Standardization of space data communication

protocol

- Establishment of multimedia data management method
- Standardization of user interface

2.3 Approach

Next, we have implemented two experiments to approach the above-mentioned technical problems. One experiment is concerning a real-time processing system that we call CS-3 (Communication Satellite type 3) experiment, and the other is concerning an off-line processing system that we call ADKAS (Advanced tamokuteki (multi-purpose, in English) data KAnri (management, in English) System) experiment.

2.3.1 CS-3 experiment

A real-time processing system experiment is relevant to the technical problem a.: standardization of data communication protocol.

In this experiment, we have adopted the AOS protocol. We use three of the different kinds of service that AOS protocol provides. The purpose of the experiment is to verify whether the AOS protocol is effective to the actual data communication.

The experiment's overview is shown in figure 2-1. In this experiment, we confirmed whether a operator who were distant from a manipulator could monitor it via CS-3. Telemetry was forwarded by the Path service of the AOS protocol, command of teleoperation by the Internet service, and the compressed image data by the Bit-stream service. The AOS Internet service is simulated on TCP/IP protocol by the application software.

2.3.2 ADKAS experiment

A off-line processing system experiment is relevant to technical problem b. and c.: establishment of multimedia data management method and standardization of user interface.

In this experiment, we developed the prototype and evaluated it. The purpose of the experiment is to verify whether the data relevant by catalog reference is applicable to multimedia data management and SFDU of CCSDS Recommendation is effective to data distribution.

The experiment's overview is shown in figure 2-2.

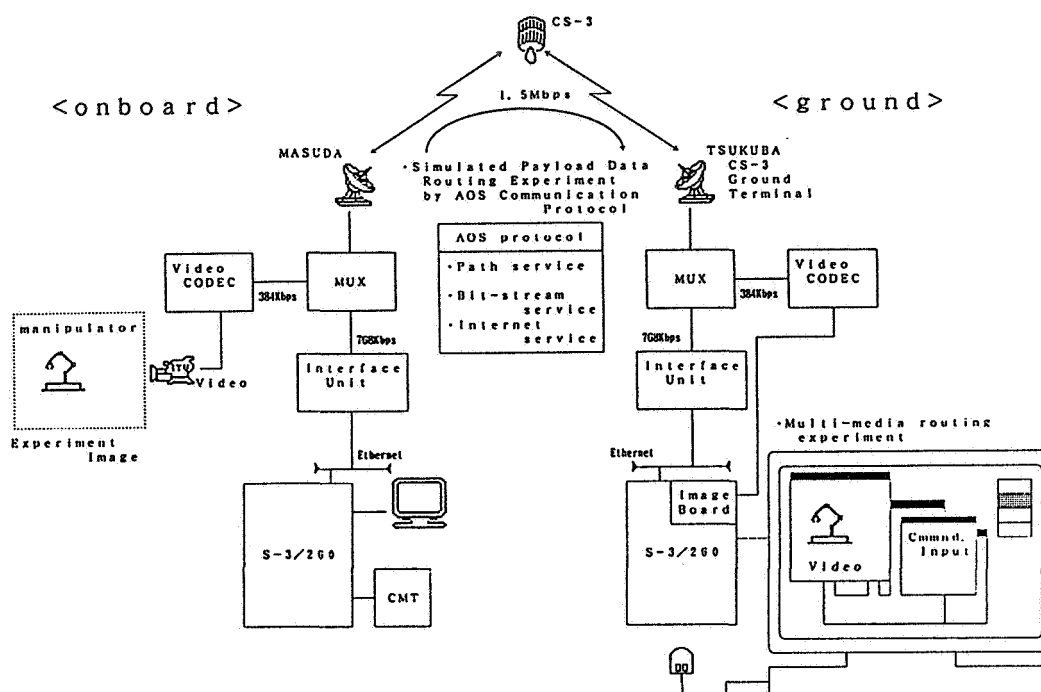


Figure 2-1: OVERVIEW OF CS-3 EXPERIMENT

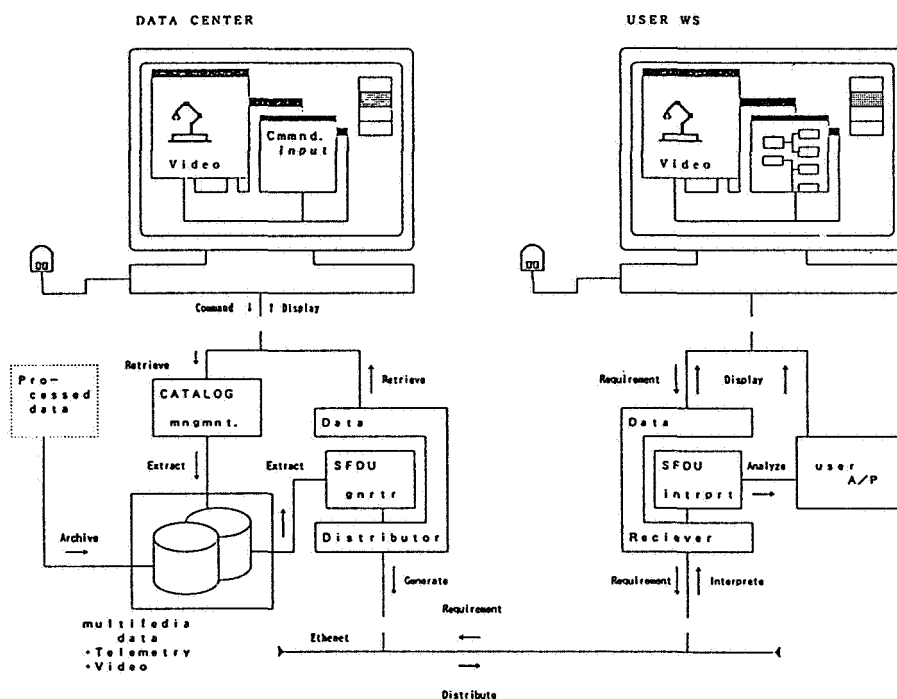


Figure 2-2: OVERVIEW OF ADKAS EXPERIMENT

We constructed the relational data base to make the catalog reference that relate each multimedia data. Moreover, we adopted SFDU, and developed the software of automatic SFDU generation/interpretation on CCSDS Recommendation.

In this paper, the report of AOS experiment concerning a real-time processing system is omitted because of introducing it in other papers. The following sections describe the ADKAS experiment concerning off-line processing system in detail.

3. Prototype

3.1 Technical elements

The ADKAS solves technical problems b. and c. (described in section 2.2). That is,
b. establishment of multi-media data management method,
c. standardization of user interface.

The technical elements in ADKAS prototype are as follows:

Technical element #1: management of the multimedia data

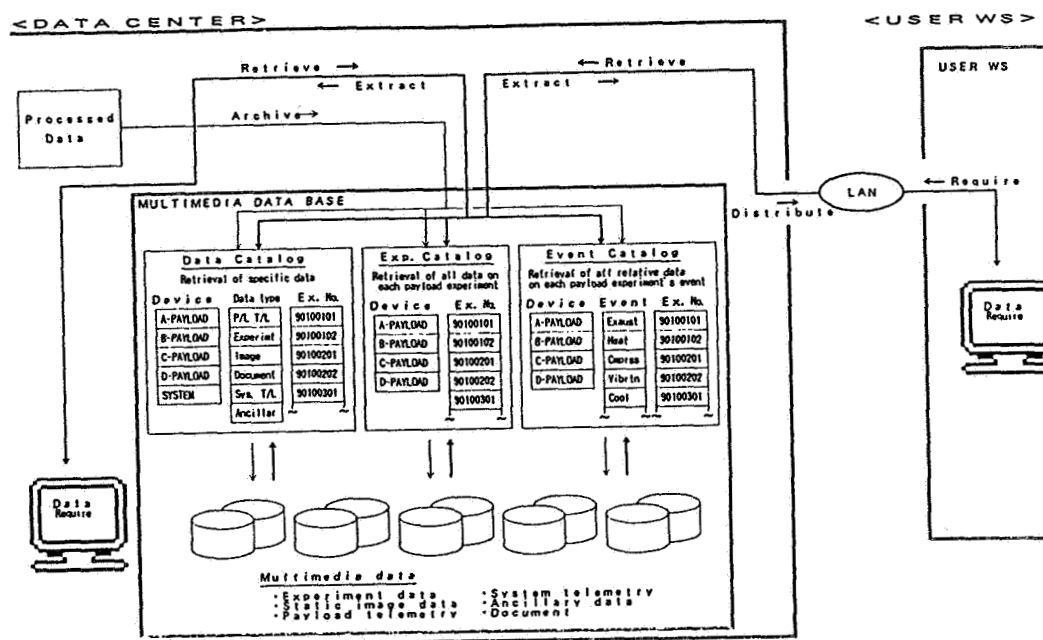


Figure 3-1: CONFIGURATION OF MULTIMEDIA MANAGEMENT SYSTEM

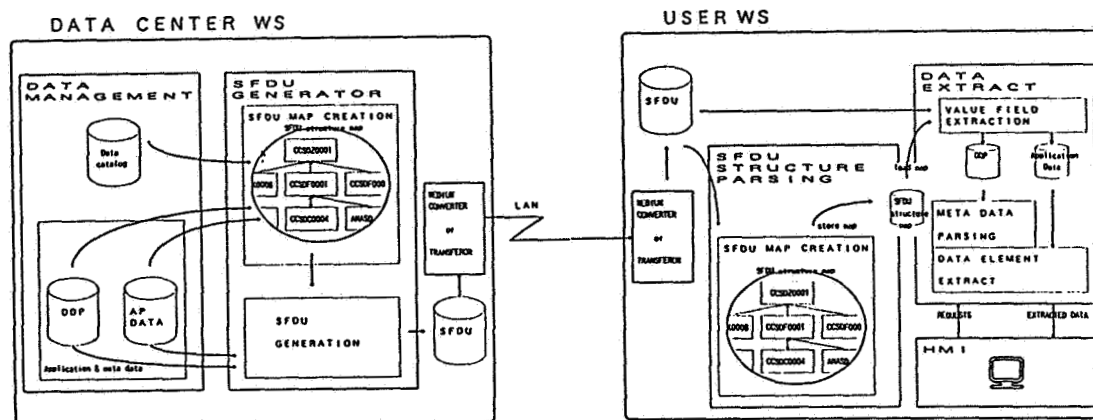


Figure 3-2: CONFIGURATION OF DATA DISTRIBUTION SYSTEM

-integrated digital data and the related analog data from the viewpoint of payload experiment or payload experiment event.

-extracted the catalog reference(constructed by relational data base) from the above-mentioned data set group.

As a result, we can retrieve all the multimedia data relevant to one another.

Technical element #2: data distribution as a lot of unspecified users require

-Developed the software of automatic SFDU generation/interpretation on CCSDS Recommendation

-Extracted SFDU generated automatically from the transmission medium and accumulated it in the multimedia data base

As a result, we can exchange multimedia data generated automatically with a lot of unspecified users by LAN or GBT

Next, we intend to describe the overview of ADKAS. The configuration of multimedia management and data distribution system is shown in Figure 3-1 and Figure 3-2.

3.2 SFDU description

In this section, we explain what is SFDU.

SFDU is defined as a method for packaging supplementary data and meta data with space related science and engineering data to create data products that contain complete sets of information for the purpose of information interchange.

Any type of data can be integrated into the SFDU domain; what is standardized is the technique of packaging together the various data objects into an SFDU data product. There is no constraint on the format of the user data.

Data instances are of little value without descriptions of their contents and organization. The SFDU concept integrates data and meta data and provides a technique to identify the different types of packaged information.

4. TESTING

This section describes the evaluation of ADKAS prototype.

4.1 Evaluations

We test the availability of the approach to the technical problems.

Evaluation #1: management of multimedia data

-Is it possible to retrieve data of each payload-experiment or payload-experiment-event?

-Is the processing time suitable?

Evaluation #2: data distribution to a lot of unspecified users in the standard format

-Is the algorithm of SFDU software suitable?

-Is SFDU able to be generated correctly?

-Is SFDU able to be interpreted correctly?

-Is the composition rule of SFDU suitable?

4.2 Results

As a result, what we learned is the following:

Result #1: management of multimedia data

-The catalog registration time becomes more long according to increasing of data number and operation-time. Moreover, the CPU load rate is increased.

-The processing time is generally proportional to the number of selected data. Result #2: data distribution to a lot of unspecified users

-The use of SFDU structural map was effective as the interface with the data management program.

-The following functions are necessary as the user interface with SFDU generator:

- Specific structure generation function; for generating SFDU by in putting parameter concerning constant composition pattern and the data source when the data is frequently exchanged and data structure need not be changed.

- Variable structure generation function; for generating SFDU by deciding SFDU structure interactively when the data is not frequently exchanged and data structure is changed.

- SFDU propriety check function; checking propriety of SFDU structure generated and updating it.

- Meta data generation function; for converting local meta data description into the standard meta data description language.

- Medium conversion function; for converting SFDU generated into the medium demanded and adding map information on the medium.

5. Conclusion

We have learned that our approaches are effective to the technical problems in Advanced Spacecraft era.

For catalog reference, it is proved that we are able to retrieve data that is synchronized

with each other, and for SFDU, it is proved that the interpretation algorithm and real-data structure are available.

It is necessary to examine the following matters so that the approach described in this paper is applied to the system in the future.

- Management of mass storage data
- Data exchange between different machines
- Automatic meta-data generation from telemetry data base
- Automatic judgement which analysis software applied to real-data in SFDU

6. Reference

[1]Standard Formatted Data Units ... structure and construction rules(May 1992)

CCSDS 620.0-B-2

Consultative Committee for Space Data Systems

[2]Standard Formatted Data Units ... A tutorial (May 1992)

CCSDS 621.0-G-1

Consultative Committee for Space Data Systems

[3]Introduction to CCSDS Panel2 (Oct 1992)

CCSDS A23.M-Y-1.0

Consultative Committee for Space Data Systems

DISTRIBUTED COMPUTING ENVIRONMENTS FOR FUTURE SPACE CONTROL SYSTEMS

N 9 4 - 2 3 9 1 1

Pierre Viallefont

C.N.E.S
Toulouse - FRANCE

ABSTRACT

The aim of this paper is to present the results of a CNES research project on distributed computing systems. The purpose of this research was to study the impact of the use of new computer technologies in the design and development of future space applications.

The first part of this study was a state-of-the-art review of distributed computing systems. One of the interesting ideas arising from this review is the concept of a "virtual computer" allowing the distributed hardware architecture to be hidden from a software application.

The "virtual computer" can improve system performance by adapting the best architecture (addition of computers) to the software application without having to modify its source code. This concept can also decrease the cost and obsolescence of the hardware architecture.

In order to verify the feasibility of the "virtual computer" concept, a prototype representative of a distributed space application is being developed independently of the hardware architecture.

Key Words: Distributed Computing, Distributed Architecture, Control Center.

1. OVERALL APPROACH

The motivation behind this research is the growing importance of distributed computing environments. First of all, a state-of-the-art review of distributed systems was made so as to reveal the main underlying concepts. We then determined what innovative contributions could be made to the design and development of our space computing applications by such distribution concepts. Once these contributions were clearly identified, it was decided to validate them by applying them to the development of a prototype representative of a distributed space application.

2. STATE-OF-THE-ART REVIEW

2.1 Concepts

2.1.1 Definition

There are many definitions of distributed systems. We chose the following one: "A distributed system is a system whose behaviour is determined by algorithms specifically designed to take into account and use several processing places".

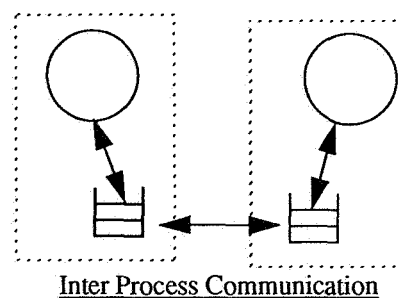
2.1.2 Client/server Model

The customer/server model is certainly the most widespread concept in the literature dealing with distributed architectures. The server defines services it makes available to the client. The client can access the server only through the set of services (functions) the server has decided to export. The server can serve several clients.

2.1.3 Distributed programming techniques

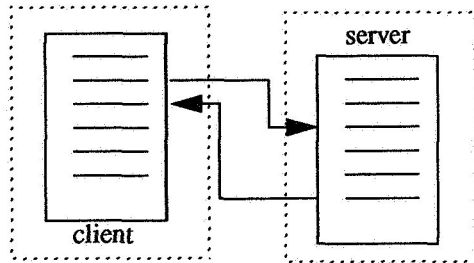
There are two categories of distributed programming techniques:

--> Inter Process Communication (IPC) allowing two remote processes to communicate by sending messages to each other. TCP/IP sockets are a good example of this.



--> Remote Procedure Calls (RPC) allowing two remote processes to communicate in a different way, namely through the transmission of parameters. The

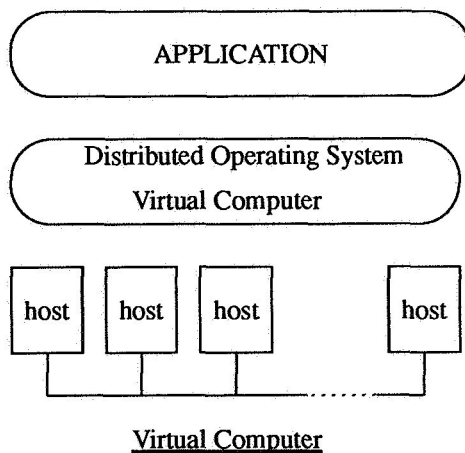
customer/server model can be implemented quite simply using RPCs. The customer calls a server function using an RPC and then waits for the answer. The server carries out the processing and returns the results to the customer. One example of the use of RPCs is that of the SUN RPCs used to build the NFS distributed file system (Ref. 1).



Remote Procedure Call

2.1.4 The Virtual Computer

Another concept highlighted in the state-of-the-art review is that of the virtual computer. This approach allows the hardware architecture to be masked to an application in order to give it the impression that it is being run on a centralized system.



The virtual computer concept is closely linked with the concept of transparency. Several transparency levels are defined in the ANSA project (Ref. 2):

--> access-to-object transparency: an object (such as a file) may be accessed (i.e. opened, read, deleted etc.) in the same way whether locally or remotely. An example of a system offering access-to-file transparency is the NFS (Network File System). However, a

real distributed system must provide this transparency for all the objects it manages (not only files, but also peripherals, processes, memory etc.).

--> location transparency: a user or an application need not worry about the location of the objects he/it is handling. The NFS also offers this type of transparency: nothing in the filenames indicates the location of these objects.

--> concurrency transparency: several users or applications may share a remote object without being aware of it.

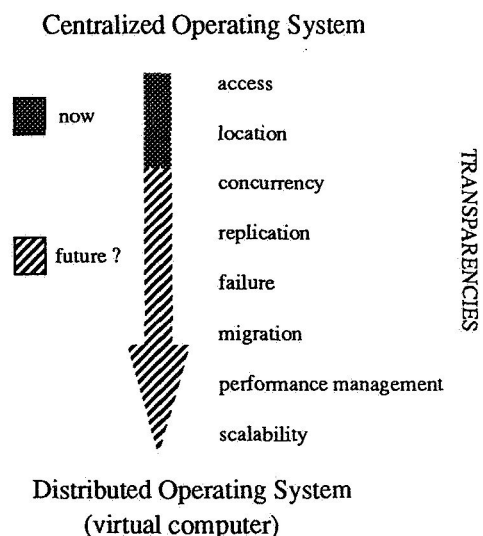
--> replication transparency: some objects are replicated without the application being aware of it. This is very useful for implementing hardware fault tolerance techniques by process replication.

--> failure transparency: the occurrence of faults is masked to applications, or at least the work in progress is completed.

--> migration transparency: objects can migrate from one computer to another without the application being aware of it.

--> performance management transparency: the system can reconfigure itself dynamically in order to improve performance in a transparent manner.

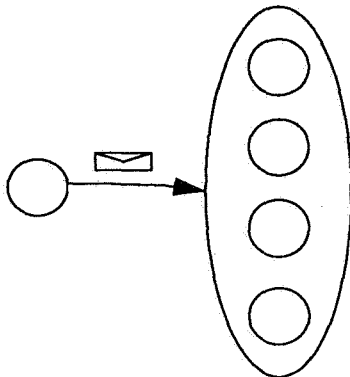
--> scaling transparency: the system or applications can change the execution scale (e.g.: increased number of computers in a network) without having to change the algorithms.



The implementation of the different transparency levels can be used to define a real distributed operating system based on the concept of a virtual computer. Most Industry or Research products provide both access and location transparency. Some provide even more, but at this point in time, none of the systems investigated are able to provide all the different kinds of transparency mentioned above.

2.1.5 Process groups

This concept can be found in many of the systems investigated. A process group, as its name indicates, groups together several different processes. Its advantage is that all the processes belonging to the same group receive the same messages.



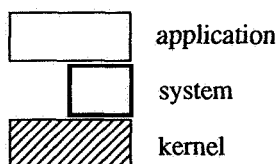
Process Groups

With this concept, message broadcast and above all fault tolerance can easily be implemented by replicating the same processes on different sites.

2.2 Systems investigated

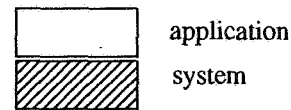
Having examined both Research and Industry products, three types of system could be identified:

--> native systems integrating distribution at kernel level. The operating system is built over the kernel. The most technologically advanced kernels at present are Chorus (Ref. 3), Mach (Ref. 4) and Amoeba (Ref. 5).



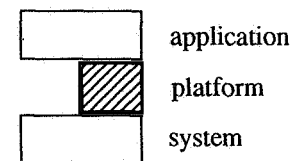
Native Systems

--> integrated systems implementing functions related to distribution.



Integrated Systems

--> platforms or toolboxes located over the operating system. They provide users with a distributed environment without masking the operating system. The most advanced are ISIS (Ref. 6), ANSA (Ref. 2), DELTA-4 (Ref. 7) and OSF/DCE (Ref. 8).



Platforms

All current state-of-the-art systems provide at least access and location transparency. Some of them offer replication transparency (like ISIS), and others fault transparency (like DELTA-4). All these systems are oriented towards use with Unix.

With respect to standardization, the OSF/DCE system would appear to be the most promising as it is supported by the majority of Unix manufacturers. Unfortunately, OSF/DCE is not yet available and will definitely not be available before 1993.

3. THE CONTRIBUTION OF THE CONCEPTS

3.1 The virtual computer

3.1.1 Adapting the architecture to the application

When building a spacecraft control center, the manufacturer and hardware architecture are both selected at the outset of the project, before software development begins. Sometimes, however, software development can last several years (1 to 5).

One never knows before the application's validation if the hardware architecture will be efficient enough to run the application. If it is not efficient enough, the current configuration either has to be upgraded (through extra memory, CPU board, additional disks etc.), or the software code has to be optimized. If the

power of the CPU board cannot be upgraded, one or more extra computers have to be added, requiring a change in the architecture, and therefore a change in the application code.

Using a distributed system implementing the concept of a virtual computer enables the architecture's distribution to be masked to the application. Thus, it is quite possible on integration to redistribute application components over another distributed architecture while maintaining its performance standards (addition of a computer) and without having to change the application code.

3.1.2 Keeping one step ahead of architecture obsolescence

Owing to the current developments in data processing technologies, the price/performance ratio of computers is constantly decreasing such that the architecture selected at the beginning of the project is technically superseded by the end, and the resulting price/performance ratio is very poor. The project investment cost may appear to be relatively high compared with the architecture's real value at the time of validation. Furthermore, there may be a better architecture/manufacturer pair at the time of the application's validation.

One solution consists in choosing the target architecture after the application has been developed. Firstly, this requires the use of a standard (Unix) operating system in order to be independent of the manufacturer. If this system implements the virtual computer concept, the application also becomes independent of the architecture. However, this poses many problems: firstly, if no architecture is chosen, on which computers will the application be developed? This problem can be solved by buying "low-quality" workstations which will be used exclusively for development work. Secondly, is it really possible to do without the specific characteristics of a project (communication protocols, fault tolerance etc.) which often determine the choice of architecture at the outset of the project?

3.2 The client/server model

3.2.1 Simplifying the development of distributed applications

The design and development of a distributed application is quite complex: using tools such as TCP/IP sockets is not easy, and the final development of a distributed application may even be distinctly difficult (error reproducibility difficulties, no final development tools etc.).

The development of distributed applications can be simplified using the RPC-based client/server model. RPCs provide interface description languages and generators which allow the developer to concentrate exclusively on the development of server and customer functions without worrying about network communication. The server interfaces are clearly defined and, as a result, their final development becomes easier.

3.3 Process groups

3.3.1 Tolerating hardware faults

Hardware fault tolerance acts as a brake upon distribution. It has a direct effect on both the architecture's design (redundant computers) and development (reconfiguration scenarios, failure processing etc.).

In some applications the problem is solved by:

--> using fault-tolerant computers (Tandem, Stratus etc.)

--> replicating computers so as to be able to reinitiate the application on the redundant computers.

The solution based on fault-tolerant computers may be deemed expensive. Computer replication may be inappropriate as it requires that the application should be stopped and then reinitiated on another machine with the same hardware configuration.

Fault tolerance problems can be solved efficiently - and without having to buy specific computers - with a distributed system implementing the process groups on replicated computers. Indeed, as the processes are replicated on distinct computers, the failure of a single computer does not affect the application's operation.

4. THE PROTOTYPE

4.1 Objectives

The objective of the prototype's development was the practical validation of the aforementioned concepts, namely the concept of a virtual computer, the client/server model and process groups.

Thus, the prototype was designed:

--> to be independent of hardware architecture. It is hoped that our application will be able to run on one,

two or "x" number of computers without having to resort to recompilation. It was therefore decided to develop and validate our distributed application on a single computer before testing it on a distributed architecture.

--> to be independent of the manufacturer by using Unix standards (portability).

--> to tolerate computer hardware failures without affecting the application's operation.

4.2 Functional characteristics

It was decided to put the previous concepts into practice in an application typical of the space environment, and building a prototype inspired by spacecraft control centers appeared a judicious idea.

The following functions were chosen:

- > Telemetry acquisition and decommutation,
- > Real-time telemetry monitoring,
- > Control and monitoring of the distributed application,
- > Real-time logbook,
- > Off-line analysis of the logbook,
- > Off-line telemetry processing.

4.3 Development environment

The development environment of the prototype consists of:

- > the ISIS toolbox developed by Cornell University, to manage fault tolerance by using the process group concept.
- > the EASY RPC product developed by the French company "Cap Gemini Innovation". This product, based on Sun RPCs, also provides both access and location transparency; it therefore enables an application to be developed independently of the architecture (access and location transparency). The EASY RPC product enables easy implementation of the client/server model, thus facilitating the programming and final development of the distributed application.
- > a Unix system complying with POSIX.1 and XPG3 standards, to be independent of the manufacturer. Whilst our application is intended to be developed on a SUN4 workstation, the target architecture will actually comprise DEC, HP and SUN Unix computers.

--> the C++ language for modular software development.

--> the OSF/Motif system for multiwindowing combined with a graphic interface generator.

5. CONCLUSION

This research project is not yet completed, since the prototype is still being developed. Nevertheless, positive results are expected and it is hoped that future developments in spacecraft control centers will integrate distribution concepts so as to be able to develop space applications which are flexible, upgradeable, hardware fault tolerant and above all independent of hardware architecture.

6. REFERENCES

1. R. Sandberg, D. Goldberg, S. Kleiman, D. Walsh, B. Lyone. 1985. "Design and implementation of the Sun Network File System". In Usenix proceedings.
2. Advanced Networked Systems Architecture. 1989. "An Engineer's Introduction to the Architecture". Architecture Projects Management Limited.
3. MP. Rouille, P. Viallefont. 1990. "Evaluation du systeme d'exploitation reparti et temps reel Chorus". Rapport de stage CNES - RA/TE/IS/MIS/SM.
4. Tevanian, Avadis, Rashid, Richard. 1987. "MACH: A basis for Future UNIX Development". Department of Computer Science, Carnegie Mellon University, Pittsburgh, Technical report CMU-CS-87-139
5. Mullender and all. 1989. "Amoeba - High Performance Distributed Computing". In Technical Report CWI, CS-R8929.
6. K. Birman, T. Joseph. 1988. "The ISIS System Manual". Cornell University, Ithaca, NY.
7. D. Powell, P. Verissimo, G. Bonn, F. Waeselynck, D. Seaton. 1988. "The Delta-4 Approach to Dependability in Open Distributed Computing Systems". In proceedings 18th Int. Symposium on Fault-tolerant Computing- IEEE.
8. Open Software Foundation. 1990. "Distributed Computing Environment- Overview and rationale".

N94-23912

DIGITAL TEST SIGNAL GENERATION: AN ACCURATE SNR CALIBRATION APPROACH FOR THE DSN

Benito O. Gutiérrez-Luaces

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, California 91109-8099, USA

ABSTRACT

In support of the on-going automation of the Deep Space Network (DSN) a new method of generating analog test signals with accurate signal-to-noise ratio (SNR) is described. High accuracy is obtained by simultaneous generation of digital noise and signal spectra at the desired bandwidth (base-band or band-pass). The digital synthesis provides a test signal embedded in noise with the statistical properties of a stationary random process. Accuracy is dependent on test integration time and limited only by the system quantization noise (0.02 dB). The monitor and control as well as signal-processing programs reside in a personal computer (PC). Commands are transmitted to properly configure the specially designed high-speed digital hardware. The prototype can generate either two data channels modulated or not on a subcarrier, or one QPSK channel, or a residual carrier with one biphasic data channel. The analog spectrum generated is on the DC to 10 MHz frequency range. These spectra may be up-converted to any desired frequency without loss on the characteristics of the SNR provided. Test results are presented.

Key Words: Signal-to-noise ratio, testing automation, space-telecommunications performance

1. INTRODUCTION

Spacecraft link performance optimization has always

been a requirement for missions supported by the Deep Space Network (DSN). This optimization relies on accurate predictions for the degradations (and losses) encountered in the different modulation and detection processes in use in the telecommunication link. Mathematical models for these processes usually are available beforehand to be later verified by tests run at the Telecommunications Development Laboratory (TDL), Compatibility Test Area (CTA-21) and sometimes at the DSN Stations. Also, the DSN has extremely high availability requirements for telemetry support of deep space missions. This requirement (as high as 99%) imposed on the Ground Data System (GDS) as a whole, immediately translates into extremely tight performance requirements on the test signals to be used as performance verification tests ("pre-track").

Parameters of a telecommunication link that need to be precisely known (Ref. 1) are: carrier signal-to-noise ratio (SNR), symbol SNR, and bit SNR for coded transmissions. Variables implicit in above measurements are: telemetry filtered wave shape, modulation index, total transmission bandwidth, orbit profiles (doppler), system noise temperature, and carrier phase jitter, as well as the telemetry and tracking receiving systems performance. Availability and performance of the entire system (GDS) is usually verified before a scheduled spacecraft pass ("pre-track tests").

These "pre-track tests" have traditionally been

performed with the well known Y-factor method. Different error sources come into play depending on whether the test is performed in the carrier or in the modulated part of the transmitted spectrum. Reported accuracy of the manual method in use today varies from 0.3 dB, in a rigidly controlled environment, to 1 dB at the DSN stations.

2. GENERAL DESCRIPTION

To improve the accuracy and reliability, and to automate the measurement process, a digital test signal generator (DTSG) has been developed at the Jet Propulsion Laboratory (JPL) to digitally synthesize the highly precise test signal. Then, the signal-to-noise ratios obtained are independent of gain variations, consequently full knowledge and control of above variables can be achieved.

The DTSG main components are: a JPL built SNR generator box (SGB) with block diagram shown in Figure 1, a personal computer (PC), and an off-the-shelf frequency synthesizer. The SGB generates signals with different spectra, including IF, dual subcarrier, or quadrature phase-shift-key (QPSK). The monitor and control, as well as other signal-processing programs, resides in the PC. The PC transmits configuration commands to the SGB, which in turn configures the special high-speed digital hardware needed to generate the output calibrated signals. A programmable frequency synthesizer generates the variable system clock (2 to 20 MHz) needed by the high-speed hardware.

Figure 1 shows in block diagram form the main functions assigned to each board residing in a Multibus-I chassis in the SGB. Three digital paths with identical hardware (pattern generator, filter, and attenuator) are used to generate two base-band filtered data channels (channels 1 and 2) of subcarrier binary phase-shift keyed data (BPSK) and one channel of base-band filtered noise ("Noise Channel") to be subsequently added to produce the analog output

$$S_T(t) = d_1(t) \sin \omega_{sc1} t + d_2(t) \sin \omega_{sc2} t + N_{BB}(t) \quad (A)$$

where: $\omega_{sc1,2}$ = the first or second subcarrier frequency, $d_{1,2}$ = the first or second base-band filtered data process, N_{BB} = the base-band filtered noise process, and $\sin x = \text{sign}[\sin(x)]$ or $\sin(x)$.

In case of residual carrier generation, channel 1 generates the carrier, channel 2 generates the modulation, and the noise channel generates the band-pass filtered noise. The filtered analog output of the DTSG in this configuration becomes

$$S_T(t) = \cos \Theta(t) \sin \omega_c t + m(t) \sin \Theta(t) \cos \omega_c t + N_{BP}(t) \quad (B)$$

where: ω_c = the carrier frequency, Θ = the modulation index, N_{BP} = the band-pass filtered noise process, and $m(t)$ = the modulation signal.

A QPSK output is similar to (A) with

$$\omega_{sc2} t = \omega_{sc1} t + \pi/2 \quad (C)$$

3. DIGITAL SIGNAL GENERATION (Ref. 1)

Referring to figure 1, the digital synthesis method consists in the generation of a gaussian digital noise ("Noise Channel") by sequentially reading a random access memory ("Data or Noise 64K RAM") at a very high speed (20 MHz). The RAM has been loaded with data bytes conforming to a quantized gaussian distribution function deduced from the analog probability function

$$f(x,u,\sigma) = 1/(\sigma\sqrt{2\pi}) e^{-[(x-u)/\sigma]^2/2} \quad (D)$$

with $u=0$ for unbiased noise and σ = standard deviation.

The sequential address to the RAM is determined by a random address generator with inherent uniform distribution and a very long period (longer than 24 hours). Therefore, the distribution function of the data bytes in the RAM determines the final distribution function of the digital noise. This digital noise is subsequently filtered by a digital filter with operator

defined frequency response. Figure 2 is an example of the frequency response of a band-pass (IF) filter. Any type of coded or uncoded data and subcarrier frequency as well as any data pattern (frames and sub-frames) can be mapped and simultaneously generated in a similar RAM ("Channel 1"). This output is subsequently filtered by an operator defined digital filter. Digital addition of these two outputs properly attenuated ("Filter and Attenuator") will provide the desired SNR. This digital result is further converted into an analog representation ("Digital-to-analog Conversion") and up-converted to the desired frequency band to be tested. The SNR of this output is truly stationary. This property is very important because long integration times can therefore be used to obtain the accuracy and precision required by a particular test. Limits will only be imposed by the inherent generator quantization noise (0.02 dB).

The DTSG will be used to calibrate SNR measurements and also losses on other signal processes. Several statistical measurements have been implemented ("Statistics Monitor"). The same statistical measurements have been implemented in the digital output as well as in the analog output.

The symbol SNR is continuously evaluated using the statistical results of the "Symbol Squared Accumulator" ($\langle S^2 \rangle$) and the "Symbol Value Accumulator" ($\langle S \rangle$) by

$$SNR_M(dB) = 10 \log_{10} \{ \langle S \rangle^2 / [2(\langle S^2 \rangle - \langle S \rangle^2)] \} \quad (E)$$

The last measurement made in this output is the symbol error count, using the statistics accumulated in the "Symbol Error Accumulator". From these symbols in error, a Symbol Error Rate (SER) is calculated. This SER is further converted into the equivalent symbol SNR and compared to the previous measured SNR_M to verify the accuracy (i.e. 0.1 dB) of the generated SNR.

In order to characterize the hardware performance and DTSG accuracy, a "Histogram Accumulator"

has been implemented providing therefore a straight forward method to confirm the actual probability density function of the filtered or unfiltered noise. Figure 3 is the histogram ($8 \cdot 10^6$ samples) of a white gaussian noise generated in the "Pattern Generator" of the "Noise Channel" after filtering by the same filter with frequency response shown in figure 2. Also displayed in figure 3 is a fit of the theoretical gaussian probability function of identical standard deviation. Note the excellent agreement of the fit with the actual filtered noise.

4. REPEATABILITY (Ref. 2)

Several readings of the digital SSNR measurement of the base-band output were obtained. Table I shows a summary of those results for different base-band configurations. The different parameters used in the test were: the base-band bandwidth (BB-BW), the subcarrier frequency (f_c), the data symbol rate (Data), the desired symbol SNR (Nom. SSNR), the measured SSNR (DTSG SSNR), and the repeatability standard deviation (σ).

Identical tests were repeated for the IF configuration case with the results summarized in Table II. In this case the variables were: the total radio frequency bandwidth (RF-BW), the intermediate frequency for the test (IF), the desired carrier SNR (Nom. Pc/No), the measured carrier SNR (DTSG-Pc/No) and the repeatability standard deviation (σ).

Note the excellent repeatability obtained in both cases and its dependency on the SNR generated (Ref. 1).

5. BASE-BAND TESTING (Ref. 2)

Several base-band configurations were tested at Goldstone's "Deep Space Station 12" by feeding the output of the DTSG configured in "Base-band Mode", to a Demodulator Synchronizer Assembly (DSA). The DSA was configured with a loop bandwidth of "Medium" and the mean SSNR ($\langle SSNR \rangle$) and standard deviation were obtained every 30 seconds. A summary of results obtained are shown in Table III.

Table III shows the capabilities of the DTSG as a SSNR source to measure telemetry equipment performance with a high degree of accuracy. Extensive testing of the different signal processing configurations in use at the DSN can now be performed with almost no test set up time. Also, existing theoretical models may now be confirmed with the needed accuracy.

6. RADIO FREQUENCY TESTING (Ref. 2)

A Digital Receiver in development at JPL was used to confirm the DTSG capabilities of accurate and repeatable radio frequency SNR generation. The DTSG 5MHz IF output (signal and noise) was up-converted to 300MHz and fed to the Digital Receiver. The DTSG was configured in the "IF Mode", providing a carrier and the corresponding telemetry modulation. The results obtained from the Digital Receiver carrier detection and the Demodulator Subcarrier Assembly (DSA) are shown in Table IV. For the DTSG configuration the variables in Table IV are: the radio frequency bandwidth used in the DTSG at 300 MHz IF (RF-BW), the measured carrier SNR (Pc/No), the subcarrier frequency (Sc), the symbol rate (Data), and the measured symbol SNR (SSNR). For the Digital Receiver configuration: the one sided loop bandwidth (B_1), and the measured carrier SNR (Pc/No) with the standard deviation of the measurement (σ). Finally, the DSA measured symbol SNR (SSNR) with the standard deviation of the measurement (σ).

Comparison of symbol SNR results of Tables III and Table IV show very good agreement. Small differences are probably due to system degradation.

Another Digital Receiver test was run with the DTSG in the "300 MHz IF Mode", a RF bandwidth of ± 150 KHz, and an unmodulated carrier. The carrier SNR (Pc/No) was decremented in 5 dB steps from 50 to 10 dB/Hz. Mean values of the Digital Receiver Pc/No measurements showed a maximum of 0.2 dB degradation. The standard deviation of the carrier SNR measurement of the Digital Receiver varied from 0.1 dB at Pc/No=50-20 dB to 0.5 dB at Pc/No=10 dB. A 0.5 dB degradation with a standard deviation of 0.6 dB was

measured at 8 dB of Pc/No. Results again show an outstanding agreement, with a probable very small system degradation.

7. STABILITY (Ref. 2)

The DTSG is intended to be used as a precise SNR test generator at any frequency band used at the DSN (S, X, or K-band). Testing capability of processes involving frequency, phase, and amplitude stabilities, such as doppler or ranging measurements and radio science experiments, should also be a desired feature.

Tests were performed with the DSN Radio Science equipment using the same testing configuration previously described in 6. RADIO FREQUENCY TESTING. Post-real-time Allan variance measurements on a strong Pc/No (approximately 50 dB/Hz) were made showing frequency stabilities very close to the expected theoretical results (10^{-15} for an integration time of 300 seconds). An amplitude stability test performed in this configuration showed a maximum standard deviation in the signal power measurement of less than 0.01 dB.

8. REFERENCES

1. Gutiérrez-Luaces, B.O. 1990. Digital Test Signal Generation: An Accurate SNR Calibration Approach For The DSN. In *TDA Progress Report 42-104*, Jet Propulsion Laboratory, Pasadena, California, USA.
2. Gutiérrez-Luaces, B.O. et al. 1992. Digital Test Signal Generation: An Accurate SNR Calibration Approach For The DSN (Part II; Performance results). To be published in *TDA Progress Report*, Jet Propulsion Laboratory, Pasadena, California, USA.

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

Table I. Base Band Repeatability

BB-BW	Sc	Data	Nom. SSNR	DTSG SSNR	σ
(MHz)	(KHz)	(S/s)	(dB)	(dB)	(dB)
4	360	43200	3	2.94	0.007
4	360	230400	3	2.90	0.002
4	960	537600	0	-0.37	0.002
0.4	32	32	0	-0.23	0.14
0.4	32	32	5	5.39	0.14
0.4	32	32	11	11.57	0.05

Table II. IF Repeatability

RF-BW	IF	Nom. Pe/No	DTSG Pe/No	σ
(MHz)	(MHz)	(dB/Hz)	(dB/Hz)	(dB)
± 4	5	50	49.84	0.01
± 1	5	30	30.00	0.01
± 1	5	17	17.36	0.12
± 0.15	5	40	40.38	0.01
± 0.15	5	20	19.59	0.10
± 0.15	5	15	14.85	0.20
± 0.15	5	8	8.14	0.28

Table III. DSA Performance verification

DTSG configuration (Standard deviation < 0.1dB)

BB-BW	Sc	Data	DTSG SSNR	DSA SSNR, $\pm \sigma$
(MHz)	(KHz)	(S/s)	(dB)	(dB)
4	360	43200	2.94	2.90 $\pm$ 0.10
"	"	"	-0.06	-0.16 $\pm$ 0.12
"	"	268800	-0.08	-0.14 $\pm$ 0.04
"	"	"	-3.12	-3.02 $\pm$ 0.06
"	968	537600	-3.36	-3.36 $\pm$ 0.06
"	0	800000	0.03	0.50 $\pm$ 0.03
"	0	1200000	0.01	1.25 $\pm$ 0.03
0.4	22.5	160	2.87	2.63 $\pm$ 0.05
"	32.8	32	10.68	10.02 $\pm$ 0.26
"	"	"	5.78	4.23 $\pm$ 0.20
"	"	"	3.09	1.25 $\pm$ 0.40

Table IV. Digital Receiver Test Results

DTSG configuration ($\sigma < 0.1$ dB)					Digital Receiver		DSA
RF-BW	Pe/No	Sc	Data	SSNR	B ₁	Pe/No, $\pm \sigma$	SSNR, $\pm \sigma$
(MHz)	(dB/Hz)	(KHz)	(S/s)	(dB)	(Hz)	(dB/Hz)	(dB)
± 4	40.5	360	268800	0.86	5	40.30 $\pm$ 0.25	0.60 $\pm$ 0.01
± 4	32.5	360	43200	1.10	5	32.20 $\pm$ 0.40	0.78 $\pm$ 0.01
± 0.15	13.0	32.8	32	3.0	1	11.60 $\pm$ 0.50	1.06 $\pm$ 0.42
± 0.15	13.0	32.8	32	3.0	0.25	12.60 $\pm$ 0.20	1.88 $\pm$ 0.25

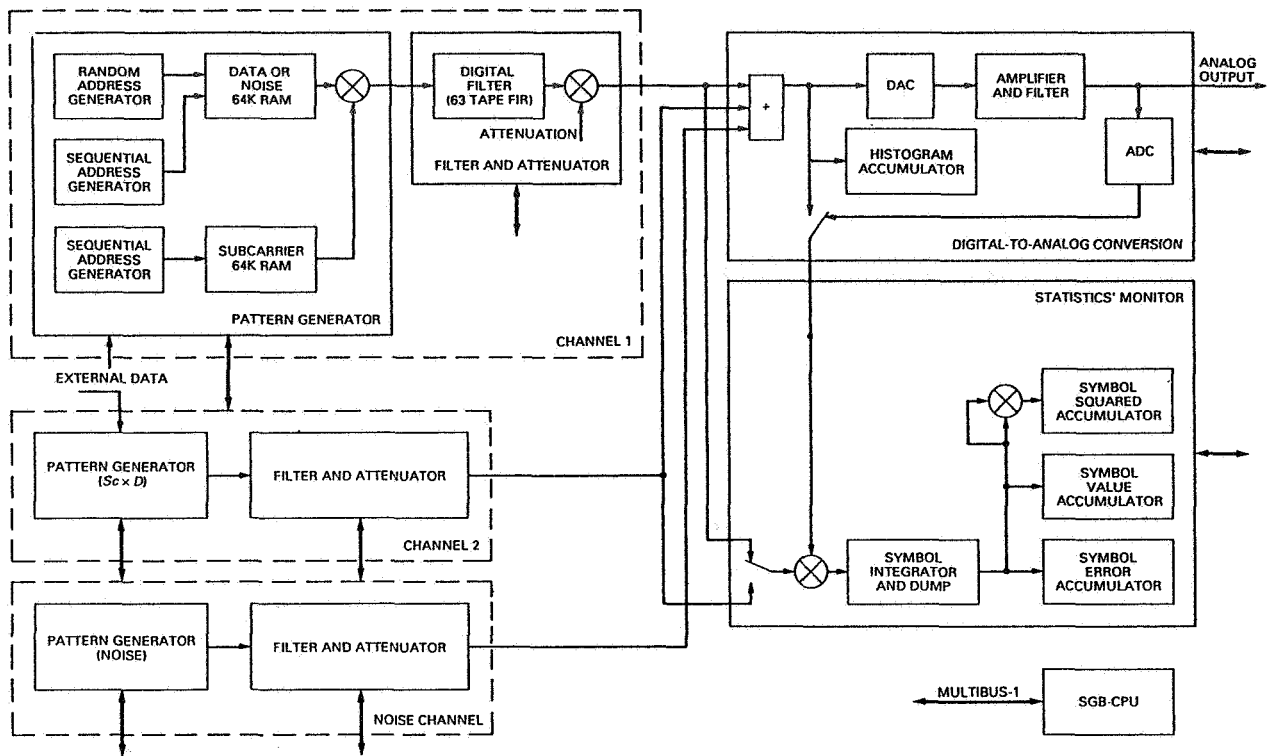


Figure 1.- SNR generator (SGB) block diagram

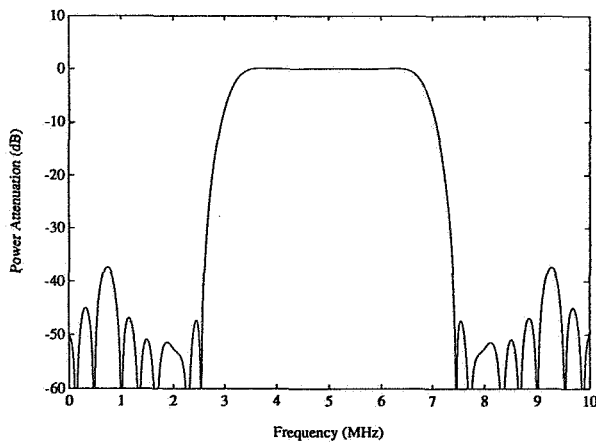


Figure 2.- Frequency response of ± 2 MHz IF filter

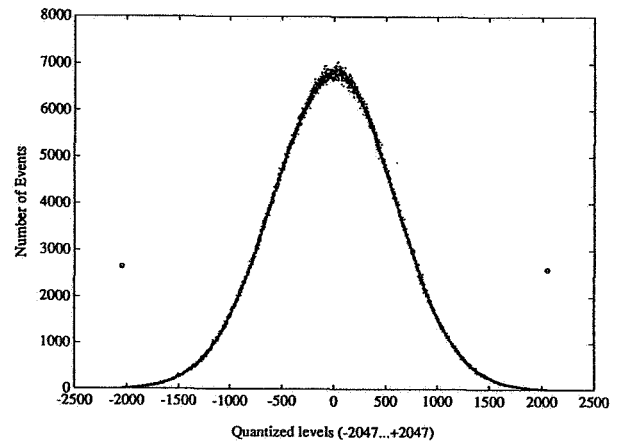


Figure 3.- Filtered discrete probability function with gaussian fit

AUTOMATIC TOOLS FOR SYSTEM TESTING

N. M. PECCIA

N 9 4 - 2 3 9 1 3

European Space Operations Centre,
Robert-Bosch-Str. 5, 6100 Darmstadt, Germany

ABSTRACT

As spacecraft control and other space-related ground systems become increasingly complex, the effort required in testing and validation also increases. Implementation of a spacecraft control system normally involves a number of incremental deliveries. In addition kernel or general purpose software may also be involved, which must itself be considered in the integration and testing programme.

Tools can be used to assist this testing. These can reduce the effort required or alternatively they can ensure that for a given level of effort, a better job is done. Great benefit could be derived by automating certain types of testing (interactive software) which up to now has been performed manually at a terminal.

This paper reports on an on-going study. The study examines means of automating spacecraft control system testing, evaluates relevant commercial tools and aims to prototype basic automatic testing functions.

Key Words: System testing, Data acquisition, automatic tools

1. INTRODUCTION

This paper gives an interim report on a study of the use of automatic and other tools for test on spacecraft control systems.

The study aims to show how this approach can shorten the time-consuming process of performing repeated tests and also increase their effectiveness.

The study examines user requirements for such automatic test driver tools, and investigates availability of suitable support software packages on the market. Based upon these requirements the architectural design for a complete test and

recording system is being produced. This design takes into account commercial ("off the shelf") software identified in the earlier part of the study. Finally a prototype of at least the basic mechanisms of the design will be developed and demonstrated.

As system under test the ESA-developed SCOS B system (Spacecraft Control and Operations System, ref. R1) is taken as a basis. SCOS-B is a reusable and configurable Spacecraft Control System Kernel. It provides the telemetry and some telecommand related functions for both classical and new packetised spacecraft data and for low orbiting, geostationary and deep space missions. The telemetry functions include real-time reception, validation, processing and filing, real time and retrieval displays, printouts and archiving. The telecommand functions include the preparation of telecommands, scheduling, validation, verification, filing, display and printouts. SCOS B's User Interface is based on SUN/UNIX workstations.

2. BACKGROUND

In ESA's experience several man-years of effort are invested in pre-launch system level tests of each spacecraft control system. Such system tests should in principle involve a realistic number of interactive "users" acting in concert according to a scenario of the actual in-orbit operations. This could be achieved by a carefully stage-managed session of testing where each participant closely follows a scripted timeline. In practice this is prohibitively expensive to the point of being impractical. Consequently, realistic tests are not carried out; at best developers will try to anticipate (or guess) usage patterns and find remaining bugs.

The tool, which is the subject of this study, is intended for testing successive releases of complex systems (non-regression testing). Non-regression testing verifies that old bugs (i.e. those already resolved in previous releases) have not been re-introduced.

3. OVERALL APPROACH

The study has the following steps:

- Problem and methodology analysis and definition of requirements.
- Evaluation of supporting tools available on the market.
- Architectural design.
- Prototyping and demonstration of the basic design.

4. TEST TOOL REQUIREMENTS

The basic requirements on the system are as follows:

- It will permit exact recording of keystrokes and mouse activities (related to the MMI of the system under test) on a file along with execution times. The multi-user system under test needs to synchronise the (controlled, simulated) user activity at each MMI.
- It will permit capture of screen images including full and partial windows. (The user can save parts of screens, full windows on the screen, and even the entire screen for later use in regression testing).
- It will permit recording of all responses from the system under test.
- It will permit editing of keystroke files (capability to add/delete/modify).
- It will be possible to pause during a recording session and to resume later.
- It will have the capability to playback the previously recorded test script, regenerating the sequence of screens.

- The user will be able to slow down or speed up the playback rate. If the playback rate is not realistic, applications which use timeouts could fail or alternatively overload could occur. This feature allows realistic matching of input data rates.
- It will permit events generated by the user during playback operation to be edited out.
- Both interactive / batch mode of operation will be possible.
- Saved keystroke/mouse-activities files can be compared.
- Saved screen files can be compared.

In summary the tool will be capable of recording all relevant input / output (keystrokes, mouse inputs, images, etc.) of the different interactive users during system level tests. Because of this, exact repeatability of tests can be realised by playback.

The Tool will serve the following uses:

1. Support to system and interaction testing of the spacecraft control in the various development stages.
2. Testing of the spacecraft simulator; in this case it would be configured with the spacecraft control system.
3. Testing the spacecraft itself (i.e. System Validation Tests SVTs). In this case it would be configured as in (1). By permitting exact reproduction of the SVT scenarios it would permit verification close out of SPRs on the S/C Control System and Discrepancy Notices (DNs) on the spacecraft.

5. EVALUATION OF

COMMERCIALY-AVAILABLE

TOOLS

5.1 EVALUATION APPROACH AND

CRITERIA

A set of typical test scenarios was defined, examining all phases from initial deliveries through to System Operations Validations (SOV).

The main requirements of the tool are described in Section 4.

Each candidate tool is evaluated against the following criteria:

- Number of requirements (section 4) satisfied.
- Product maturity, qualifications and status.
- Price and availability.
- Hardware requirements consumption.
- Availability on different hardware / software platforms.
- Results of stress and function tests done on each tool.

The first step for the evaluation is to identify the set of candidate tools. This implies to conduct a market survey.

The first criterion is the most important. The score against this requirement increases with number of requirements satisfied. Also to arrive at this score, the requirements are assigned as follows:

- *Essential:* The tool is rejected if any essential requirement is not fulfilled.
- *Important:* Fulfilment of each important requirement adds a large increment to the score against criterion 1.
- *Useful:* Each such requirement satisfied will add a small increment to the score.

For each tool, the extent to which each requirement is met will be allocated to three levels: total, partial and null; where null means the tool does not address the requirement concerned. For partially met essential requirements, the effort needed to develop additional software is evaluated.

5.2 TOOLS EVALUATED

The following candidate tools were identified:

- XTM (USA, Public domain)
- STW / REG, Software Test Works™, (Software Research Inc.), California, USA
- VALID^R (Datamat Ingegneria dei Sistemi, S.p.A.), Italy.
- gTSimX™ (Siemens Nixdorf AG), Germany
- preVue-X™ (Performance Awareness), USA
- X Runners™ (Mercury Interactive Corporation)

5.3 CONCLUSIONS

To date, only STW/REG and VALID have been evaluated, with results reported below:

5.3.1 STW / REG

Advantages:

- Independent of the application under test
- Screen oriented. It captures and executes user events anywhere in the entire screen (root window) identifying for each event all the relevant information.
- Powerful image management. It provides utilities for the capture and comparison of partial or total results, represented by screen images. Furthermore, it fully supports the masking of the images.
- It provides a set of utilities comprising from Keysave editing facilities, preview options,

image view to files comparison facilities for different types of files.

Disadvantages:

- Images dump and comparison consumes a lot of CPU time
- No support of multi-user interaction
- No user intervention during playback

5.3.2 VALID

Advantages:

- Application oriented. The events captured are relative to the object and not to the absolute coordinates on the screen
- It has a powerful test language for the development and delivery of test procedures.
- User intervention during playback is possible.
- Support of multi-user interaction
- Powerful tool for different stages of the software life cycle (Software Requirement Phase, Architectural design phase, etc.)

Disadvantages:

- It is oriented through graphical objects (widgets), disabling the possibility to store arbitrary parts of the screen.
- No relative time between actions captured, so properly timed playback is not possible. Although "pause" statements may be included in the test files.
- High Consumption of CPU

Based upon these evaluations and considering that both tools are more complementary than exclusive, it was decided to use both tools as the basis for the development of the prototype.

6. IMPLEMENTATION OF THE PROTOTYPE

The prototype is to be implemented on a SUN 4 workstation (Openwindows 3.0, SUN OS 4.1.2) actually used by the software support team on the EURECA project. Some of the spacecraft control subsystems will be tested using both tools (STW / REG and VALID) capturing the different events of the test session in parallel. This will permit comparison of the two tools.

7. CONCLUSIONS

At the time of writing this paper (November 1992) the Architectural Design phase is in progress.

This study aims to produce a prototype which can be used in the system testing of control systems for upcoming ESA missions such as those of CLUSTER, PASTEL and ARTEMIS.

8. REFERENCES

1. "Architectural Design Document for Spacecraft Control and Operation System", SCOS Functional System (SFS), Issue 1, Revision 02, March 1987.
2. "ESA Software Engineering Standards", ESA PSS-05-0, Issue 2, February 1989.
3. "Software Test Works - Test Regression Tools", Release 4, 1991, Software Research Inc.
4. VALID, Reference Manual, Version 1.3 C, 1992, Datamat, Ingegneria dei Sistemi S.p.A.
5. VALID, User's Guide, Version 1.3 C, 1992, Datamat, Ingegneria dei Sistemi S.p.A.

REPRESENTING OPERATIONS PROCEDURES USING TEMPORAL DEPENDENCY NETWORKS

Kristina E. Fayyad and Lynne P. Cooper

Monitor & Control Technology Group
M/S 525-3660Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA 91109
kristina@isd.jpl.nasa.gov
lpcooper@ai.jpl.nasa.gov

N94-23914

ABSTRACT

DSN Link Monitor & Control (LMC) operations consist primarily of executing procedures to configure, calibrate, test, and operate a communications link between an interplanetary spacecraft and its mission control center. Currently, the LMC operators are responsible for integrating procedures into an end-to-end series of steps. The research presented in this paper is investigating new ways of specifying operations procedures that incorporate the insight of operations, engineering, and science personnel to improve mission operations. The paper describes the rationale for using Temporal Dependency Networks (TDNs) to represent the procedures, a description of how the data is acquired, and the knowledge engineering effort required to represent operations procedures. Results of operational tests of this concept, as implemented in the LMC Operator Assistant Prototype (LMCOA), are also presented.

Keywords: Automation, Knowledge Engineering, Operations

1. INTRODUCTION

DSN Link Monitor & Control (LMC) operations consist primarily of executing procedures to configure, calibrate, test, and operate a communications link between an interplanetary spacecraft and its mission control center [Ref. 1]. The procedures can be organized according to two taxonomies: those based on the individual subsystems which form the link (e.g. antenna, receiver, telemetry processor), and those specific to the spacecraft or science experiment. Currently, the LMC operators must manually integrate the individual subsystem and higher level spacecraft procedures into an end-to-end series of steps needed to support realtime operations. On-the-job experience provides the necessary background for determining any interdependencies between subsystems, for interpreting mission specific

configuration and performance requirements for routine, often-performed operations, and for determining the fastest, most reliable means of getting the job done. Non-routine operations, however, often require operators to perform different types of operations and to interpret results in ways different than they normally would. For example, during an often performed pass (e.g. telemetry), the primary goal is to maximize the signal quality. Operators may adjust the various equipment in order to improve the signal-to-noise ratio (SNR). For a radio science type pass, performed much less frequently however, the configuration of the equipment must remain constant and the SNR may actually be part of the data collected.

The purpose of our current research is to investigate new ways of specifying operations procedures that enable automated operations, capture the insight and expertise of operations, engineering, and science personnel, and lead to more efficient and reliable operations. [Ref. 2] The goals for the representation are:

1. Extensible: it must be capable of representing the full spectrum of operations procedures.
2. Flexible: it must allow for the variations in operations procedures experienced between the different operations complexes, and under different circumstances.
3. Maintainable: because the station equipment and types of operations are constantly changing, it must be easy to update and maintain.
4. Verifiable: the representation must be testable for accuracy.
5. Robust: it must provide the information necessary to:
 - identify problems
 - perform workarounds
 - interpret monitor data
 - tailor a general procedure to specific circumstances

6. User Natural: it must be readable and usable by both computer and human operator without translation.

2. APPROACH

Our approach is to use a temporal dependency network (TDN) to represent LMC operations procedures. A TDN is a directed graph which incorporates temporal and behavioral knowledge and also provides optional and conditional paths through the network. The directed graph (or, Petri Network) represents the steps required to perform an operation. Precedence relationships (step A has to happen before step B) are specified by the nodes and arcs of the network. The behavioral knowledge identifies system-state dependencies in the form of pre- and post- conditions. Temporal knowledge consists of both absolute (e.g. Acquire the spacecraft at time 02:30:45) and relative (e.g. Perform step Y 5 minutes after step X) temporal constraints. Conditional branches in the network are those performed only under certain conditions. These are the IF (this condition) THEN (do/don't do that action). The conditionals are used primarily for error recovery. Optional paths are those which are not essential to the operation -- but may, for example, provide a higher level of confidence in the data if performed.

The TDN representation is discussed in more depth in a following section. In order to understand the representation it is necessary to first understand the basic building block of the TDN, directives.

2.1 Directives

The primary data unit of the TDN is a directive. A directive is a control message which is sent to an individual subsystem in order to perform a specific function. The primary data fields are the intended subsystem, the control action, and any associated parameters. To support the TDN model of

procedures, we use a more detailed representation of the directives. Each directive is represented as an object containing the following information. An example of a directive definition is presented in Table 1.

1. Directive syntax (subsystem, message name, required and optional parameters).
2. The function of the directive: what primary and side effects it has on the subsystem; what changes it causes in any devices or subassemblies.
3. Parameter definitions: any constraints on the parameters and the support data used to determine parameter values.
4. Directive responses: the response messages sent from the subsystem to the LMC to acknowledge receipt of the directive. This is only a communications handshake and does not indicate that the directive was successfully executed.
5. Rejection notices: messages sent by the subsystem when the directive has failed to execute. (Includes syntax errors as well as realtime failures).
6. Monitor and event information: data that may be generated by the subsystem based on the actions of the directive. Specifies which parameters and user interface displays to monitor to confirm that the directive has successfully executed.
7. Pre-conditions: what state must the system be in before this directive can be sent.
8. Post-conditions: which state the system is in when the directive has successfully executed.

The information in the directive definitions is stored in a knowledge base. Of the above listed types of information, only a subset, dealing primarily with syntax and general responses, is available in the DSN documentation. Much of the information is available

Actions	Transmits predict data set CW to the ACS
Sources	Log DOY-067-1991, line 189
Preconditions	Predicts available
Postconditions	Predict table is filled Predicts downloaded successfully
Responses	COMPLETED. PROCESSING DLOAD REQUEST
Rejections	COMPLETED. INVALID PREDICT SET NAME
Event notice messages	PA 14:INTERPOLATING CW SUB1 ... PA 14:ACS CONFIRM DLOAD... PA 14:ACS <time>

Table 1. Directive Example, AP DLOAD PRED CW

only through the operations personnel (monitor information, pre- & post- conditions) or the engineers (Side effects, pre- & post- conditions).

2.2 Temporal Dependency Network

A TDN is a complex object which encodes the information necessary to perform a specific operational task. As described earlier, the primary representation of the TDN is an augmented directed graph. In the graph, each arc represents a strict precedence relationship, each node a sequence of directives which perform a subset of the overall function. The network explicitly specifies the precedence relationships between nodes, any potential parallelism, and rules for recovering from global faults. The nodes, or blocks, consist of the directives, temporal constraints, pre- and post-conditions, and local recovery information should the block fail. An example block is given in Figure 1.

Once the directives necessary to perform the given operation and any pre- and post- conditions specific to the type of pass are known, designing a TDN becomes an exercise in assigning directives to the correct block. Preconditions specify device states that must be true before the directive can execute. Postconditions specify the expected device states after the directive has successfully executed. Precedence relationships in the TDN are formed by ensuring that the actions required to satisfy a directive precondition occur and are verified before it executes. So, if two directives are in sequence because one depends on the successful completion of the other, these directives will be placed in separate blocks and a precedence relationship formed between them.

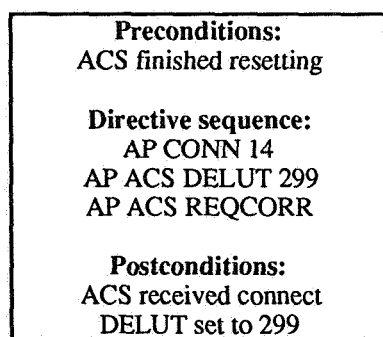


Figure 1. Block Example

Directive preconditions are pushed up to the block level, so that before the block begins executing its first directive, all preconditions of all directives in that block must be satisfied. In some cases, this check is redundant because completion of the previous block is dependent upon satisfying a

postcondition which satisfies the precondition of the next block. We have designed the TDN in this way for two reasons: 1) the block may, at some point be moved to a different location in the TDN, and 2) if a device fails between the end of the first block and the start of the second, we have a way to detect the failure before proceeding.

The TDN is the general representation of an operational task. An instance of the TDN is created from the general representation and parameterized for the specific pass being performed. From this perspective, the TDN acts as a template for operations, and individual parameters (time, frequency, file names) are filled in at execution time to perform operations. A sample of a general TDN, shown at the block level, is given in Figure 2.

3. KNOWLEDGE ENGINEERING

There are two knowledge bases required for the TDN. The first is the TDN itself, which is a high-level procedural representation. The second is the Directive Dictionary which contains the definitions for all of the directives used in our test TDN. An overview of the knowledge engineering activities used to create both knowledge bases is given in the following sections. For the purposes of this paper, we define knowledge engineering to be the process of acquiring, representing, testing, and validating the knowledge in a knowledge base. In our case, the knowledge engineering process was tightly coupled to the design process, and the results of our knowledge engineering efforts influenced overall system design of the LMCOA Prototype.

Common to both knowledge engineering efforts was the need to keep detailed information as to the sources of information. The process of gathering knowledge is iterative and the sources may be referred back to multiple times. For example, the information from different personnel and documentation may be contradictory, so it may be necessary to refer back to each source to determine why. A significant part of our knowledge base development was dedicated to documenting the source of information, the date, and subsequent changes.

3.1 TDN Knowledge Engineering

The TDN concept for procedure representation is being tested by creating a TDN to support precalibration and data collection for a Very Long Baseline Interferometry (VLBI) Delta Differential One-Way Ranging (DDOR) pass at the Goldstone 70 Meter Antenna. Our first step in representing the VLBI DDOR was to create a baseline sequence of

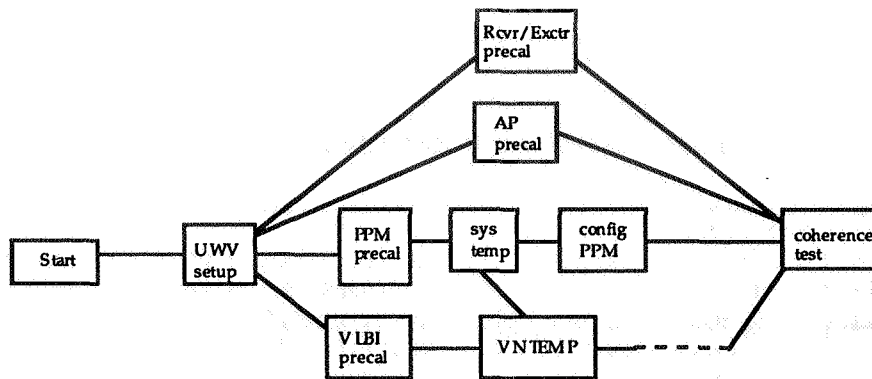


Figure 2. TDN Example, Block level

directives and other actions that the operators execute. While acquiring this information from various operators and engineers, it became clear that there is no single sequence that defines this activity. Since several subsystems are necessary and a significant part of their setup is independent of the other subsystems, the representation had to incorporate the network features of the TDN from the start to represent this inherent parallelism. In addition, other variations in the procedures would arise when problems were detected and steps taken to resolve them during actual operations.

The primary source for information concerning the TDN was through people. In order to effectively interview and acquire knowledge from the operations personnel, we needed a basis for discussion. We began by pulling all of the directives from a log of a sample pass. We created a draft TDN, in the form of individual directives (no blocks at this point) using post-it notes on (a 22 ft. length of) butcher paper¹. The operations personnel marked up the TDN and moved around the different directives. All comments resulting in changes or additions to the TDN included the source for the information. This proved extremely useful, especially when contradictions in input arose.

The contradictions themselves were a form of knowledge. They represented a wide variety of causes including: 1) A difference between what the scientists wanted and what operations personnel thought they wanted; 2) Procedure data that was valid only as a work around until a system bug was corrected; 3) differences in how the engineers designed the subsystems and how the operators actually use them; 4) Errors in the documentation. The TDN knowledge is still in the process of being

validated by review with operations personnel and both laboratory and on-site testing.

3.2 Directive Dictionary Knowledge Engineering

The directive dictionary contains the directives for the subsystems necessary to perform a VLBI DDOR: antenna, microwave, receiver/exciter, precision power monitor, metric data assembly, and VLBI/signal processor. The knowledge that is collected for a single directive consists of: directive syntax and parameters, actions of the directive, directive responses, event notice messages and monitor data which may be generated based on the actions of the directive, and rejection notices. In addition, it is necessary to find out what subsystem devices these directives affect and are being referred to in these various messages. The original source for the majority of the directive information was the operations manuals for the individual subsystems. However, the majority of the effort expended in generating the Directive Dictionary was spent filling in the gaps and correcting inconsistencies in the manuals by using other sources. For example, LMC operator logs were used to identify directive responses and some of the event notice messages that can be expected.

Identifying the preconditions necessary before a directive can successfully execute and the post-conditions after it executes required the collection and cross-correlation of many pieces of information. Each subsystem may consist of several devices and subassemblies. In many cases, the directive is sent to the subsystem controller which in turn sends a control message to the appropriate device. In these circumstances, the assistance of the engineering staff was essential to determining what the desired effect of the directive really was and what information to monitor to determine that it had, in fact, executed. Some precondition information was available from the manuals in the form of device states which must

¹ To our knowledge, this is the first time a complete end-to-end sequence for an operational procedure has ever been fully documented.

Knowledge Source	Knowledge Obtained
Operations Manuals and Operations Procedures	Directive Syntax Directive Responses Procedural Overview
Logs	Sequence of directives Sample responses Sample monitor data and event messages Sample anomalies
LMC Operators	Pre- & Post- conditions Support data Sequence of directives What they actually monitor, displays used System quirks Anomaly recovery actions False alarms and "noisy" feedback
Operations Engineers	Desired operations sequences Pre- & Post-conditions Preventative actions Subsystem quirks
Design Engineers	Device and subassembly reactions Side effects
Scientists	Desired sequence of "actions" (the events that should take place, rather than the specific directives) What's important to the success of the experiment Rules for improving data quality

Table 2. Summary of Knowledge Sources

be true, or other directives which must have been successfully completed before executing a specific directive. The actions of the preceding directive can be translated into a pre- or post- condition. Precondition information was also gathered from evaluating event notice message data which specified error conditions when a directive failed.

3.3 Knowledge Engineering Summary

The knowledge engineering effort for the VLBI DDOR procedure currently consists of approximately 110 directives and 45 blocks. The sources used for knowledge engineering and the types of information obtained from those sources are listed in Table 2. The main thrust of the knowledge engineering effort has taken place over the past year. During that time, the DSN operational stations have undergone several modifications and updates to operational software. A major challenge in the knowledge engineering effort has been not just to capture the knowledge, but to also keep it current. For example, during our effort, the recommended precalibration procedure outlined by the antenna operations engineer has changed over 8 times.

4. STATUS

The knowledge engineering and TDN development activities have been performed in support of the LMC Operator Assistant Prototype (LMCOA) advanced development effort. In September 1992, the LMCOA had reached the point where its functionality and design [Ref. 3, 4] could be tested in an operational situation. After successful compatibility and functionality testing, we began testing the accuracy and completeness of the knowledge bases necessary to demonstrate automated² operations. These tests have included evaluation of the information in the Directive Dictionary in addition to end-to-end procedure testing of the TDN. In preliminary tests, the TDN has proven successful in meeting the goals outlined in Section 1 of this paper. A full scale demonstration of the LMCOA is scheduled for December 1992. The LMCOA is implemented on a NeXT Workstation using Objective C, Interface Builder, and CLIPS.

² To be precise, it will be semi-automated operations. The microwave subsystem requires manual configuration and there are actions, such as a safety page to warn people that the antenna will be moving, which the operator is required to perform.

5. ACKNOWLEDGMENTS

The work described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. The other members of the LMCOA team are Lorraine Lee, Randy Hill, Sanguan Chow, Kathy Sturdevant, and Juan Urista. We would like to acknowledge the contributions of the many operations and engineering personnel at JPL and the Goldstone Deep Space Communications Complex.

6. REFERENCES

1. Cooper, Lynne P., Rajiv Desai and Elmain Martinez, "Operator Assistant to Support Deep Space Network Link Monitor and Control," SOAR Symposium, Houston, TX, 1991.
2. Cooper, Lynne P., "Operations Automation Using Temporal Dependency Networks," Technology 2001, San Jose, CA, December 3-5, 1991.
3. Hill, Randall W., Jr. and Lorraine F. Lee "Situation Management in the Link Monitor and Control Operator Assistant (LMCOA)", SpaceOPS 92: Second International Symposium on Ground Data Systems for Space Mission Operations, Pasadena, CA, 1992.
4. Lee, Lorraine and Randall W. Hill, Jr., "Process Control and Recovery in the Link Monitor and Control Operator Assistant," SOAR Symposium, Houston, TX, 1992.

Situation Management in the Link Monitor and Control Operator Assistant (LMCOA)

Randall W. Hill, Jr. and Lorraine F. Lee

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Drive
Pasadena, CA, 91109-8099, USA
MS 525-3631 & 525-3660

N94-23915

ABSTRACT

This paper describes a knowledge-based system called the Situation Manager that was developed for the Link Monitor and Control Operator Assistant (LMCOA) at the Jet Propulsion Laboratory. This system was developed in response to a number of deficiencies that were identified in an earlier version of the LMCOA: (1) the need to close the control loop between sending a directive and knowing when its execution is complete (versus just closing the communications loop), (2) the need to recognize an anomaly and alert the operator when a directive is rejected or a link device fails, and (3) the need to suggest ways to work around an anomaly, provided that it is recognizable. In response to these needs, the Situation Manager has been designed to provide the LMCOA with three basic capabilities: situation assessment, anomaly diagnosis, and recovery from commonly occurring problems.

Key Words: Monitor and control, knowledge-based system, operator assistant, Deep Space Network, automation

1. INTRODUCTION

The goal of the Link Monitor and Control Operator Assistant (LMCOA) is to increase the availability of the Deep Space Network (DSN) by reducing the amount of time it takes to configure, calibrate and operate the communications links used to support deep space communications missions (Ref. 1). The LMCOA seeks to achieve this goal by decreasing the cognitive and motor load that is currently placed on LMC operators (Ref. 2,4). This paper describes the Situation Manager, the knowledge-based system at the heart of the LMCOA. The Situation Manager supports the LMCOA in the following ways: (1) it monitors device states, (2) it evaluates the appropriateness of control actions with respect to the device situation, and

(3) it diagnoses and recovers from simple situational anomalies.

To illustrate the need for an Operator Assistant, consider the following: during a typical pass an LMC Operator interacts with five different subsystems consisting of fifty or more devices, where each device has multiple attribute-values. In all, the operator monitors 250 or more unique data elements on subsystem displays and reads 500 or more event notice messages indicating significant state changes in subsystem devices. While performing the task, the operator issues 200 or more directives to control subsystem devices and gain access to subsystem data, and for each directive issued a response is returned that indicates whether it was accepted or rejected by the subsystem. As a consequence of the current system design, a large portion of the operations time is spent just configuring and calibrating the communications link, which is becoming increasingly more unacceptable in light of the increased demand for DSN availability for mission support.

In the next section we describe the Device Model, which is the Situation Manager's internal representation of the communications link being monitored. Following the discussion of the Device Model, we discuss how control actions are modeled in the Situation Manager. The Action Model was derived through extensive interviews with LMC operators and engineers, and it is encapsulated in a representation called the Temporal Dependency Network (TDN) (Ref. 3). Finally, we describe how the Situation Manager uses the knowledge about devices and control actions to support monitor and control activities being implemented in the LMCOA.

2. DEVICE MODEL

One of the primary tasks performed by LMC operators is to monitor the state of the devices in the communications link. The Situation Manager performs this function for the LMCOA

Device	Attribute	Value
RECORDING-DEVICE	NAME	LD0
	SELECTED?	YES
	MODE	RECORD

Table 1. Device Model Representation

by keeping an attribute-value model of all of the relevant link devices. An attribute is a property of the device that can be monitored or evaluated. An attribute-value represents the actual state of a particular device property.

2.1 Representation

A device is any entity in the communications link that has an observable state. Thus, the Device Model includes physical entities such as the Phase Calibration Generator, the Digital Tone Extractor and the Recording Devices, and it also includes software (e.g. the Narrow Channel Bandwidth (NCB) Program and the Predicts File). Each device has a name and a variable number of attributes and values. A typical device representation is shown in Table 1.

2.2 Sources of information

The LMCOA routes data about the state of the link devices to the Situation Manager's Device Model from several different sources: event notice messages, subsystem monitor data, directive responses and user input. Event notice messages describe significant state changes in the devices being monitored and they are always sent to update the Device Model. Likewise, directive responses are always routed directly to the Situation Manager, though they do not always provide useful state information. Monitor data, on the other hand, is not sent directly to the Device Model. Rather, the Situation Manager requests this data from the LMCOA's monitor database only when it is needed. This policy avoids the problem of inundating the Situation Manager with frequent updates which often reflect no change in value. The Situation Manager may also request user input for device state information when other information sources fail.

3. ACTION MODEL

Whereas the Device Model is used to represent and monitor events in the communications link, the Action Model represents the knowledge of when to send directives and how to know whether they have had their desired effects. The Action Model is represented in a framework called the TDN that explicitly encodes knowledge about procedures, directives and their situational

dependencies, thereby providing a pass-specific, integrated, end-to-end operational sequence of action.

We determined the need for the TDN by studying the behavior of several different expert LMC operators performing a particular type of pass called Very Long Baseline Interferometry Delta Differential Oneway Ranging (VLBI Delta DOR). We found that there was much more to their operational expertise than merely following the procedure manuals. Though explicit directive sequences are given in the procedure manuals, an expert operator typically brings additional knowledge to bear on the task. Specifically, the experts have an implicit knowledge about directive preconditions and postconditions. In other words, the operator knows a directive will only have its desired effect when the devices are in a particular state (i.e., preconditions), otherwise the directive will be rejected, or worse, it will have a deleterious effect on the target device. Once a directive is issued, the operator knows what effects to expect in the devices (i.e. postconditions), and if the effects are not observed then something is wrong. Thus, the expert LMC operator combines situational awareness of the communications link devices with knowledge about directive preconditions and postconditions.

In addition to having knowledge of directive preconditions and postconditions, expert LMC operators also have an implicit knowledge of how to order the procedures they perform. The procedure manuals are typically subsystem oriented, but since the task entails interacting with many subsystems the LMC operator must choose an order in which to execute the procedures. There is no overall description of the operational sequence which optimally depicts how to execute the procedures, so we find that the same task goals can be achieved by different orderings among procedures and directives. Moreover, we find that LMC operators attempt to optimize the overall task by interleaving procedures for different subsystems whenever possible.

3.1 TDN Blocks as Default Procedures

One of the goals of the TDN is to optimize task

Label	Directive	Device	Attribute	Value
PRE1	NLOAD JK	VLBI-PREDICT-DATA	RECEIVED?	YES
PRE2	NLOAD JK	VLBI-PREDICT-DATA	NAME	JK
PRE3	NLOAD JK	NCB-PROGRAM	MODE	IDLE
POST1	NLOAD JK	VLBI-PREDICT-DATA	LOADED?	YES

Table 2. Representation of Directive Preconditions and Postconditions

performance by interleaving procedure execution. One of the ways that this is achieved is by using the TDN block, which is the primary unit of organization in the TDN. A TDN block contains a sequence of directives, where there are no interdependencies among the directives within the block. The execution order among TDN Blocks is defined by a pairwise precedence relationship. This precedence relationship is transitive and creates a partial order for the directives in the affected blocks. If no order is specified between two blocks, either explicitly or implicitly by transitivity, procedural interleaving is achieved by executing the independent partial orders of directives in parallel. Otherwise, a block of directives must successfully execute before its successor block is allowed to begin its execution.

3.2 Directive Preconditions and Postconditions

Once a block of directives is eligible for execution (i.e. when its successor block is done), the situation must be evaluated to determine whether the current block's directives can be safely issued. The situation is evaluated using directive preconditions and postconditions, which represent a situational relationship between a directive and one or more devices. Each directive may have multiple preconditions and postconditions. For instance, in Table 2 the "NLOAD JK" directive has a precondition labeled PRE1 that says the predicts file must be present (i.e., RECEIVED? YES) in order for the directive to be accepted. In addition, the predicts file must have the name "JK", since this is the parameter specified by the NLOAD command. Finally, NLOAD also has the less obvious precondition, labeled PRE3, which states that the Narrow Channel Bandwidth Program must be in the IDLE mode (rather than in the RUN mode). The postcondition for this directive, labeled POST1, states that the subsystem must give an indication that the predicts file was successfully loaded once the directive has been issued.

4. SITUATION MANAGER

The Situation Manager provides cognitive support to the LMCOA by monitoring and

evaluating devices in the communications link and by making situated control decisions for the TDN Execution Manager. To accomplish this the Situation Manager uses two kinds of knowledge: mission-specific and operational. Mission-specific knowledge is represented declaratively and it includes the Device and Action Models previously described. Operational knowledge is represented using productions that match against the mission-specific knowledge and perform different tasks. In this section we discuss how the mission-specific knowledge is used by the Situation Manager's operational knowledge to perform the following functions: (1) process input messages, (2) update the Device and Action models, (3) evaluate the situation, (4) diagnose anomalies, and (5) recover from anomalies.

4.1 Flow of Control and Data

The Situation Manager interacts extensively with other LMCOA modules (Figure 1). In this section we give a detailed account of these interactions in order to provide a context for describing the individual functions within the Situation Manager.

The Situation Manager loads mission-specific Device and Action models during the initialization phase. Once a mission begins, the TDN Execution Manager analyzes the TDN and chooses a set of TDN blocks to execute based upon the precedence relations among the blocks. Prior to issuing the directives contained in the blocks, however, the TDN Execution Manager consults with the Situation Manager about whether all of the preconditions are satisfied for the directives in the selected blocks. This consultation takes place in two phases. First, the TDN Execution Manager sends a copy of each of the parameterized directives to the Situation Manager, which uses the parameter information to update the preconditions in its Action Model. Second, it requests the Situation Manager to evaluate the preconditions for each of the TDN blocks. If the preconditions for a block of directives are all satisfied, the Situation Manager gives permission for the

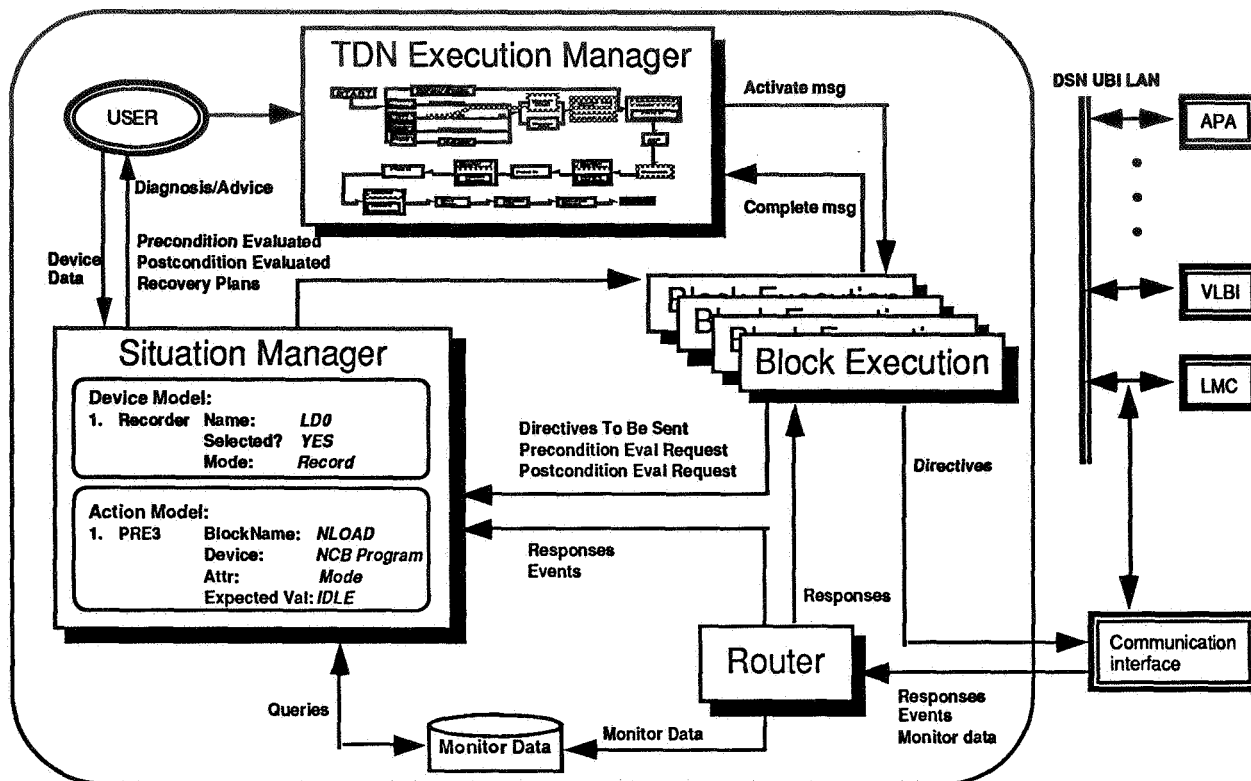


Figure 1. Data and Control Flow in LMCOA

block to be executed, and the directives are issued to the subsystems in the communications link. If any preconditions are not satisfied, the Situation Manager suspends the block's execution and sends a message to the user interface indicating that there is a problem with a specific directive associated with a particular device attribute. The message indicates both the actual and the expected device attribute values, and the user is expected to respond by either fixing the problem that was noted in the message, or by providing the current actual value for the Device Model. The latter action assumes that the user has access to information that is more current than the Situation Manager's Device Model. Once the Situation Manager sees a response to the precondition error message, it re-evaluates the block's preconditions. If the same precondition is still unsatisfied, the user has the option of overriding the precondition so that the block's directives will be issued in spite of the Situation Manager's warnings. Once a block's directives have been issued, the TDN Execution Manager requests the Situation Manager to evaluate whether the block's postconditions are satisfied (i.e. check whether all the block's directives executed as expected). The

postcondition evaluation proceeds in much the same way as the precondition evaluation.

4.2 Processing Input Messages

Each time the Situation Manager is invoked it first enters a phase where it processes input messages. It uses knowledge base rules to recognize the different types of messages and parses them into forms usable by subsequent phases. The message types handled by the system include event notice messages, directives, directive responses, directive rejections, user-responses, user-override-responses, and requests for precondition or postcondition evaluation.

4.3 Updating Device and Action Models

The Device and Action models are initialized with a set of default values from their mission-specific knowledge bases. The models change rapidly over time in response to changes in the communications link and adjustments in the procedures. The Device Model is affected primarily by event notice messages and directive responses. Each event notice message type has a specialized rule that reads and interprets the

Device	Attribute	Old Value	New Value
VLBI-PREDICT-DATA	RECEIVED?	NO	YES
	NAME	NULL	VN

Table 3. Device Model Change Caused by an Event Notice Message

Label	Directive	Device	Attribute	Old Value	New Value
PRE2	NLOAD JK	VLBI-PREDICT-DATA	NAME	JK	VN

Table 4. Modification to a Precondition via Directive Parameterization

message string and subsequently updates the appropriate device in the model. To illustrate this point, consider a message that is generated after the predicts set has been received by one of the subsystems: PRED SET VN RECEIVED (008). This message causes the modifications to the Device Model shown in Table 3.

Similarly, when the TDN Execution Manager sends a directive to the Situation Manager the Action Model is modified to reflect the parameters used with the directive. For instance, if the directive "NLOAD VN" was sent to the Situation Manager, the precondition we previously defined in the Action Model would be modified to reflect the "VN" parameter as shown in Table 4. Thus, by sending the Situation Manager a copy of the parameterized directive, the affected precondition for the directive is updated with the correct parameter value.

4.4 Evaluating Preconditions and Postconditions

When asked to evaluate the preconditions (or postconditions) of a TDN block, the Situation Manager retrieves all of the block's directives and their associated preconditions. The precondition specifies an expected value for a particular device attribute. Thus the evaluation consists of comparing the expected device attribute value with an actual device attribute value. If the two values are the same, then the evaluator records that the precondition is satisfied and continues the evaluation with the next precondition. If the two values are not the same, then it immediately diagnoses the problem and invokes a simple recovery scheme that will be described below. Control is returned to the evaluation procedure once the asynchronous recovery scheme is invoked, and the evaluation continues until all of the preconditions have been processed. If there are one or more unsatisfied preconditions, the block will be suspended and re-evaluated once the user has taken corrective action.

4.5. Diagnosing anomalies

There are three basic types of anomalies: (1) pre/postcondition failures, (2) directive rejections, and (3) event notice alarm messages. Precondition and postcondition failures occur when the actual device state is not equal to the expected state for a particular directive. The diagnosis in this case consists of the following steps: (1) isolate the device attribute for which the evaluation failed, (2) verify that the associated value in the device model is current by polling the monitor database, and (3) if steps 1 and 2 still indicate an unsatisfied pre/postcondition, notify the user with an asynchronous diagnostic message containing the device name, attribute, and its expected and actual values. The other two types of anomalies, directive rejections and event notice message alarms, have standardized explanations associated with them that are retrieved by the knowledge base's diagnostic rules.

4.6. Recovering from anomalies

The Situation Manager helps the operator to recover from pre/postcondition failures by providing diagnostic information about the failure in a dialogue window. The message contains the TDN block name, directive name, device name, attribute, and the expected and actual values. Recall that the actual value comes from the Device Model and may not reflect reality. Hence, recovery begins when the operator compares the Device Model actual value with reality for the indicated device attribute. If the Device Model's actual value does not match reality, the operator updates the value in the dialogue box. If the Device Model's actual value is accurate, then the operator indicates this is so in the dialogue box. In this way, the Situation Manager insures that the Device Model is current and re-evaluates the pre/postconditions of the suspended block. If preconditions are still not satisfied after being evaluated a second time, the

user can override the precondition, thereby breaking out of a potential loop. Otherwise, if the operator chooses not to override the unsatisfied precondition the cycle of diagnosis and recovery will be repeated, and the operator can attempt to resolve the disparity by issuing separate commands to the subsystem in question to place the device in the expected state.

The Situation Manager helps the operator to recover from directive rejections and event alarms by providing a narrative description of the directives that should be issued in order to solve the problem. Future work will focus on how to dynamically construct a recovery block of directives to deal with all three anomaly types. Though the current system assists the operator in recovery by providing diagnostic information and other descriptive narrative, it does not dynamically compose directive sequences to solve problems.

5. SUMMARY

In attempting to improve the operability of the LMC system, we seek to reduce the cognitive load that is currently placed on LMC operators. This is accomplished by providing automated support to both the monitor and the control activities performed by the operator. The LMC Operator Assistant Situation Manager maintains a dynamic model of communications link devices and uses this model to evaluate the preconditions and postconditions of control actions. In cases where the preconditions or postconditions are not satisfied, the Situation Manager provides specific feedback to the operator about why an action should not be taken or why it failed.

6. ACKNOWLEDGEMENTS

The work described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. The other members of the LMCOA team are Sanguan Chow, Lynne Cooper, Kristina Fayyad, Kathy Sturdevant and Juan Urista. We would like to acknowledge the contributions of Pam Wolken, Terry Dow, Dave Girdner, Elmain Martinez and the many operations personnel at Goldstone Deep Space Communications Complex who have shared their time and expertise with us in this effort.

7. REFERENCES

1. Cooper, Lynne P., Rajiv Desai and Elmain Martinez, "Operator Assistant to Support Deep Space Network Link Monitor and Control," SOAR Symposium, Houston, TX, 1991.
2. Cooper, Lynne P., "Operations Automation Using Temporal Dependency Networks," Technology 2001, San Jose, CA, December 3-5, 1991.
3. Fayyad, Kristina E., and Lynne P. Cooper, "Representing Operations Procedures Using Temporal Dependency Networks", SpaceOPS 92: Second International Symposium on Ground Data Systems for Space Mission Operations, Pasadena, CA, 1992.
4. Lee, Lorraine and Randall W. Hill, Jr., "Process Control and Recovery in the Link Monitor and Control Operator Assistant," SOAR Symposium, Houston, TX, 1992.

Using Mach Threads to Control DSN Operational Sequences

N94-23916

Juan Urista

Monitor & Control Technology Group
 Jet Propulsion Laboratory
 California Institute of Technology
 4800 Oak Grove Drive
 Pasadena, CA, 91109-8099, USA
 MS 525-3660

ABSTRACT

The Link Monitor and Control Operator Assistant prototype (LMCOA) is a state-of-the-art, semi-automated monitor and control system based on an object-oriented design. The purpose of the LMCOA prototyping effort is to both investigate new technology (such as Artificial Intelligence) to support automation and to evaluate advances in information systems toward developing systems that take advantage of the technology (Ref. 1).

The emergence of object-oriented design methodology has enabled a major change in how software is designed and developed. This paper describes how the object-oriented approach was used to design and implement the LMCOA and the results of operational testing. The LMCOA is implemented on a NeXT workstation using the Mach operating system and the Objective-C programming language.

Key Words: Mach, thread, monitor and control, operator assistant, object-oriented programming, Deep Space Network.

1.0 Background on LMCOA Prototype and Functional Description

The Link Monitor and Control Operator Assistant (LMCOA) prototype demonstrates semi-automated monitor and control for a Deep Space Network (DSN) station. The LMCOA consists of four major modules: a TDN Execution Manager, a Situation Manager, a Router and a Monitor Data Base. The LMCOA also contains multiple, concurrently executing TDN Block

modules. The TDN Execution Manager is responsible for traversing the TDN and starting execution of each TDN Block at the appropriate time. Each TDN Block verifies preconditions and postconditions with the Situation Manager and sends its internal list of directives sequentially. The Situation Manager maintains a device model to track the predicted and actual state of the system. It provides closed loop control, anomaly detection and recovery assistance. The Router is responsible for traffic between the LMCOA and the other modules. The Monitor Data Base stores subsystem data for future querying by the Situation Manager.

The LMCOA prototype uses a temporal dependency network (TDN) to represent LMC operations procedures. A TDN is a directed graph that incorporates temporal and behavioral knowledge and provides optional and conditional paths through the network. The directed graph (or, Petri Network) represents the steps required to perform an operation. Precedence relationships (step A has to happen before step B) are specified by the nodes and arcs of the network. The behavioral knowledge identifies system-state dependencies in the form of pre- and post-conditions. Temporal knowledge consists of both absolute (e.g., Acquire the spacecraft at time 02:30:45) and relative (e.g., Perform step Y 5 minutes after step X) temporal constraints. Conditional branches in the network are those performed only under certain conditions. These are the IF (this condition) THEN (do/don't do that action). The conditionals are used primarily for error recovery. Optional paths are those which are not essential to the operation -- but may, for example, provide a higher level of confidence in the data if performed (Ref. 2).

2.0 LMCOA Object-Oriented Design

Object-oriented design is a methodology for managing data and the functions that act upon the data. The data and its functions are known as a complex object. Major principles for decomposing complex objects into a manageable form include classification, encapsulation and inheritance. Classification is a mechanism for identifying similar things. For example, in the LMCOA, individual directives can be classified into a general directive class because all directives share basic traits such as a destination subsystem, a source subsystem and a command with parameters. Even though the value for these elements may vary for specific directives, the attributes of a directive remain the same. The specific elements of a class, where the attributes are filled with some value, are called objects of a class.

Encapsulation is a principle for combining data and specific methods that act on the data into one entity -- a class. A user of a class sees only the methods for interaction -- the detailed data remains hidden. This results in simple and modular interfaces to system modules that behave in very specific ways. Furthermore, encapsulation eases the task of code maintenance and upgrading. The internals may be upgraded and as long as the external interface remains the same -- the users of the object will never need to know about the upgrade. The upgrade is virtually invisible to the users of the object (Ref. 6,7).

Inheritance is another feature of an object oriented development environment. It is a mechanism for expressing similarities among classes. For example, in the LMCOA, there are 'control directives' and 'display control directives.' Control directives are issued to the antenna and associated subsystems. Display control directives are issued to the monitor and control system in order to bring up displays. In the LMCOA, we define a general class called 'directives' and then create a subclass called 'control directives' which inherits most of its behavior from the more general 'directives' class.

2.2 Mapping of LMCOA Architecture Into Objects

A TDN is a complex object that encodes the information necessary to perform a specific operational task. As described earlier, the primary representation of the TDN is an augmented directed graph. In the graph, each arc represents a strict precedence relationship, each node a sequence of directives that perform a subset of the overall function. The network explicitly specifies the precedence relationships between nodes, any potential parallelism, and rules for recovering from global faults. The nodes, or blocks, consist of the directives, temporal constraints, pre- and post- conditions, and local recovery information should the block fail.

The TDN block object contains the methods that read the directives and check the block pre- and post- conditions. These conditions notify the block whether the directive has been satisfied. The block is activated by the TDN class and provides a mechanism to allow concurrent processing. Each active block is a thread that is interleaved by the operating system kernel. The blocks cycle through its execution reading directives and checking its pre- and post-conditions. Once the block has completed, conditions are established for the next block to be executed. These conditions notify the TDN that activates the next block. The Directive object contains the directive string obtained from the DSN. This instance contains the data necessary to format and create the DSN block. The DSN block, known as the DSN Standard Subsystem Block or SSB, is the data format required for DSN subsystem communications (Ref. 4).

The Router object contains the methods that provide the communication path between the DSN and the LMCOA. The Router consists of a socket connector and a listener. The socket connector uses UNIX 4.3 BSD sockets to establish communications to the DSN network. The socket is connected to a communications program residing on an IBM compatible Personal Computer (PC 286AT) installed with an IEEE communications board that is required for communications with the DSN network. The communications program reads data from the DSN interface network unit and routes the data through a PC ethernet socket. The listener is

spawned as a thread and checks the socket port waiting for data to arrive.

The Situation Manager object is responsible for updating the action model. The updates are based on the directives being sent and the responses and event notification messages being received (Ref. 3).

3.0 Implementation of Object-Oriented Design

Each major module in the design is represented as an object. For example, the LMCOA defines the objects Directive, Block, and TDN. Each object contains the necessary data and behavior actions used by the LMCOA. Every time a directive is issued, it is an instance of the directive object. The block object is responsible for issuing the directive, while the TDN manager controls which blocks are active. The major modules of the LMCOA are shown in Figure 1.0. The modules correspond to objects used by the LMCOA.

The execution of the LMCOA begins with the TDN object. The TDN object identifies which block object is ready to begin execution. Once active, the block sequentially sends its list of directives to the DSN subsystems. The block is active until all the block's postconditions are satisfied. This is achieved when the DSN subsystems respond to the directives, issue data to the LMCOA through directive responses and/or event notice messages, and the Situation Manager verifies that the directives executed correctly. The responses and event notification messages are obtained through the Router object, which sends the data to both the block objects and the Situation Manager object.

3.1 General Description of Threads

A thread is an operating system construct which is the basic unit of process execution and scheduling. Each thread carries only the portion of the processor state that is necessary for independent execution. Thus threads must reside within a normal UNIX process which carries the entire processor state. However, multiple threads can execute concurrently within the context of a single process. Multiple threads, executing within a process, share the same memory space.

A change in shared data by one thread can be seen by all other threads in the process (Ref. 5). In a multi-processor environment, concurrently executing threads on separate processors can vastly improve execution time. In a single processor environment, threads are extremely useful for logical concurrency such as when several actions need to be taken on the data at the same time. For example, the major LMCOA modules are implemented as threads for simultaneous execution on the same data from the DSN subsystems. In the LMCOA, threads are also useful for interleaving directive execution and as a way to default scheduling of concurrent components to the operating system.

3.2 How Threads Were Used in LMCOA

Before the TDN Execution Manager begins the execution cycle, the LMCOA loads the TDN files and spawns the Situation Manager and the Router. The Situation Manager thread executes and begins by loading the knowledge base. After which, the Situation Manager waits for input from other modules. DSN monitor data blocks are routed to the LMCOA monitor database for later retrieval by the Situation Manager.

The LMCOA design uses multiple threads to achieve the scheduling and execution of directives. In the LMCOA, the directives are grouped according to functions known as TDN Blocks. The LMCOA begins execution by spawning the TDN object as a thread. This thread is the execution manager of the LMCOA. The operator starts the TDN execution by selecting the start button that invokes the "startTDN" method. This method invokes and spawns the start TDN Block object as a thread. The TDN start block starts the TDN blocks as parallel executing processes. The parallel block objects are spawned as threads and the operating system kernel time-slices each thread. Time-slicing the threads allows the LMCOA to execute each of the threads as interleaved processes. The LMCOA defaults the execution of the threads by allowing the kernel to schedule the threads in a logical and fair concurrent order.

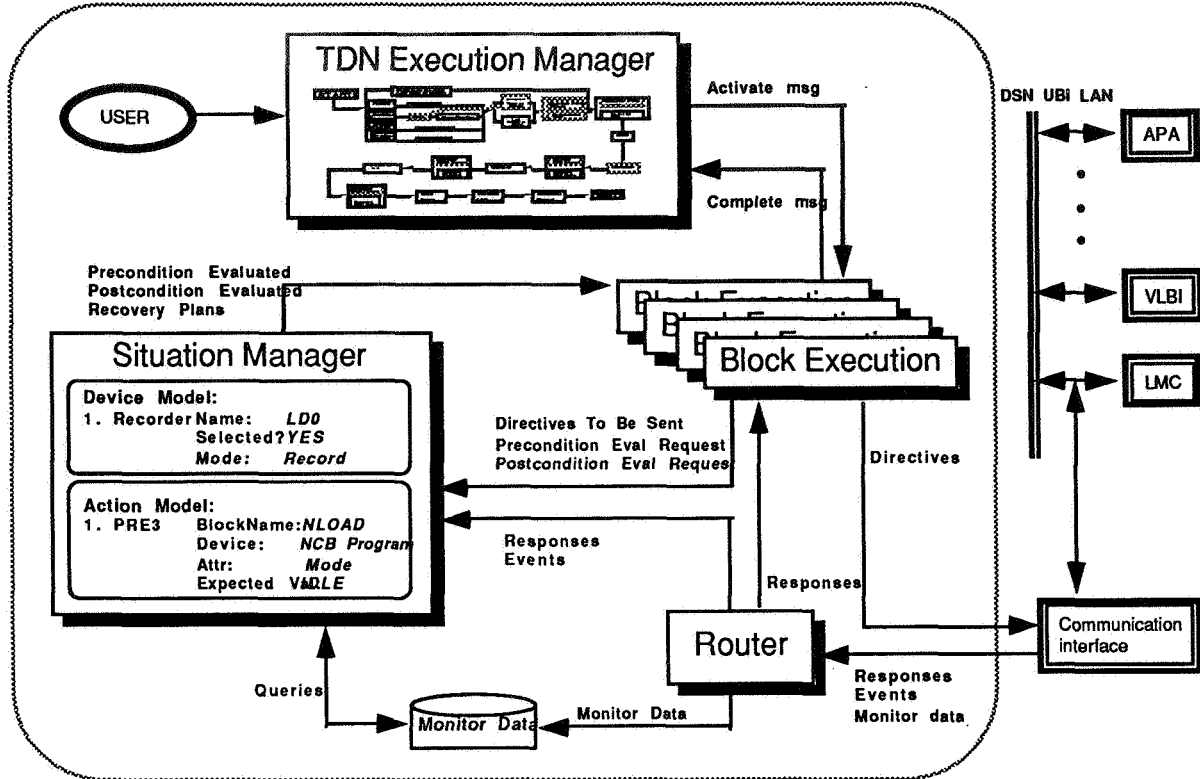


Figure 1.0 LMCOA Major Modules and Control Flow

The blocks start the process of sending directives by invoking the directive object. The directive object encapsulated methods are used by the block thread. A block can have a number of directives executing or waiting to be executed. When the block is ready, it sends the directive to the DSN subsystem. Figure 2.0 illustrates some of the threads that are spawned by the LMCOA.

Each of these threads remains within the same memory space of the LMCOA process. The threads are released when they are done executing. As shown in Figure 2.0 the TDN Execution Manager thread spawns block threads "Load APA Predicts" and "APA Precal" in parallel with block threads "Load VLBI Predicts", "VLBI Precal", and "PPM Precal." These block threads are implemented in the LMCOA as precalibration directive sequences for the DSN subsystems

Antenna Pointing Assembly (APA), Very Long Baseline Interferometry (VLBI), and Precision Power Monitor (PPM).

3.3 Example

To illustrate how threads operate in the LMCOA, Figure 3.0 shows a portion of the LMCOA Temporal Dependency Network. Table 1.0 shows a typical life cycle of the block threads shown in Figure 3.0. The TDN, Router, and Situation Manager threads span the entire program life cycle of the LMCOA process. Blocks, for example block "Load APA Predicts" execute only during a portion of the time. This block thread formats and sends directives to the DSN Antenna Pointing Assembly subsystem. Concurrently, the "Load VLBI Predicts" block thread is executed which, in turn, formats and sends the directive to the DSN Very Long Baseline Interferometry subsystem.

These threads execute in parallel and are able to share the same memory space. Thus allowing for an efficient use of both data sharing and module execution.

4.0 Status and Results

In September of 1992, the object oriented implementation of the LMCOA was tested at the Goldstone Deep Space Communications Complex. The LMCOA operated the 70-meter antenna in parallel with the existing LMC subsystem. The LMCOA object oriented design and use of the Mach operating system allowed the operator to issue parallel directives to the Antenna Subsystem and the Very Long Baseline Interferometry Subsystem. The LMCOA successfully issued the directives and processed the directive responses and subsystem event notification messages. This test allowed us to evaluate the design and Mach operating system and the results indicated that the use of objects and threads lends itself well into the designing systems that require concurrent and parallel execution. Additional testing is continuing on the knowledge base and the Situation Manager.

5.0 Acknowledgments

The work described in this paper was performed by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration. The other members of the LMCOA team whose contributions help support this work are Sanguan Chow, Kristina Fayyad, Randall W. Hill Jr, and Kathy Sturdevant. Special thanks to Lynne P. Cooper and Lorraine Lee whose encouragement, patience, and enthusiastic support made all this work possible.

6.0 References

1. Cooper, Lynne P., Rajiv Desai and Elmain Martinez, "Operator Assistant to Support Deep Space Network Link Monitor and Control." SOAR Symposium, Houston, TX. 1991.
2. Fayyad, Kristina E., and Lynne P. Cooper, "Representing Operations Procedures Using Temporal Dependency Networks", SpaceOPS 92: Second International Symposium on Ground Data Systems for Space Mission Operations, Pasadena, CA., 1992.
3. Lee, Lorraine and Randall W. Hill, Jr., "Process Control and Recovery in the Link Monitor and Control Operator Assistant," SOAR Symposium, Houston, TX. 1992.
4. JPL Internal Document 890-131, Deep Space Network DSCC General Data Flow Standards, Version 1.0, Revision C., November 14, 1990.
5. NeXT Operating System Software, NeXT Computer, Inc, 1990.
6. Coad, P., and E. Yourdon, *Object-Oriented Analysis*. Yourdon Press, 2nd Edition, 1991.
7. Boach, G., *Object-Oriented Design With Applications*. Benjamin Cummings, 1991.

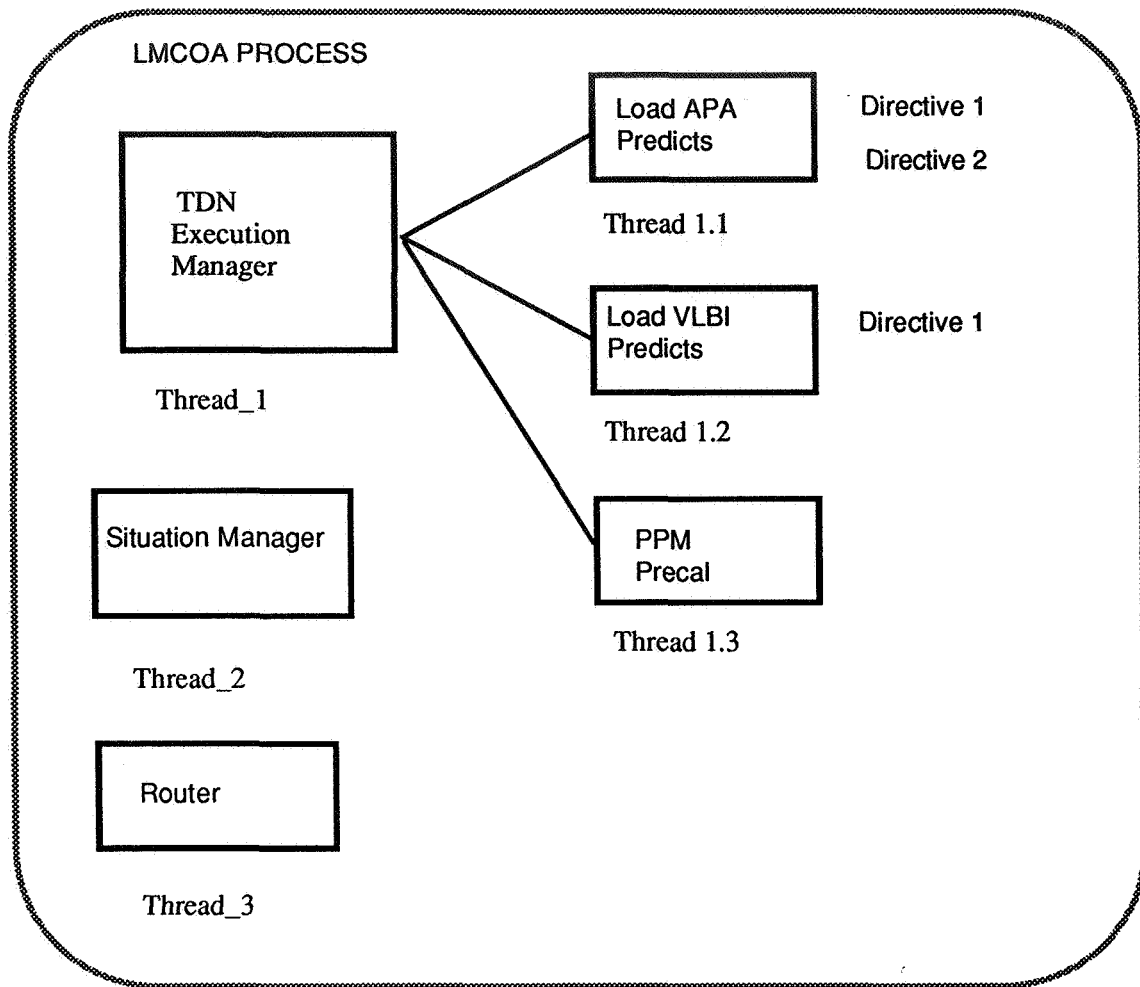


Figure 2.0 LMCOA Object as Threads

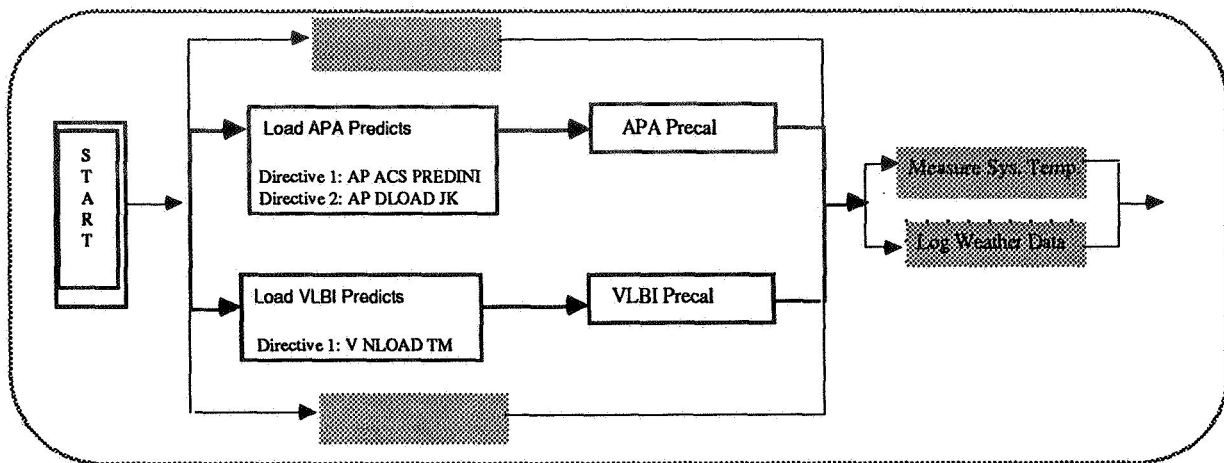


Figure 3.0 LMCOA Partial Temporal Dependency Network Representation

Object	Time --- LMCOA Process Execution Cycle
TDN	
Router	
Situation Manager	
"Load APA Predicts"	
Directive 1: AP ACS PREDINI	
Directive 2: AP DLOAD PRED JK	
"Load VLBI Predicts"	
Directive 1: V NLOAD TM	
"APA Precal"	

Table 1.0 LMCOA Process and Thread Life Cycles

N94-23917

DEEP SPACE NETWORK (DSN), NETWORK OPERATIONS CONTROL CENTER (NOCC) COMPUTER-HUMAN INTERFACES

Alvin Ellman and Magdi Carlton

California Institute of Technology
Jet Propulsion Laboratory
Pasadena, California, USA

ABSTRACT

The Network Operations Control Center (NOCC) of the DSN is responsible for scheduling the resources of DSN, and monitoring all multi-mission spacecraft tracking activities in real-time. Operations performs this job with computer systems at JPL connected to over 100 computers at Goldstone, Australia and Spain. The old computer system became obsolete, and the first version of the new system was installed in 1991. Significant improvements for the computer-human interfaces became the dominant theme for the replacement project. Major issues required innovating problem solving. Among these issues were: How to present several thousand data elements on displays without overloading the operator? What is the best graphical representation of DSN end-to-end data flow? How to operate the system without memorizing mnemonics of hundreds of operator directives? Which computing environment will meet the competing performance requirements? This paper presents the technical challenges, engineering solutions, and results of the NOCC computer-human interface design.

Key Words: computer-human interfaces, control center, automation, operations

1. INTRODUCTION

The Network Operations Control Center (NOCC) of the DSN is responsible for scheduling the resources of DSN and monitoring all multi-mission spacecraft tracking activities in real time. Operators monitor network performance and identify, isolate and correct network problems. This is done from workstations at JPL connected to over 100 computers worldwide. The old system was failing to meet the users' needs, required modernization and needed redesign to allow for growth. A replacement project was begun in 1988, and the first release of the new system was

implemented in 1991. Significantly improving the computer human interface became the dominant theme of the replacement project. However, the project team was faced with problems. There was no standard methodology in place for operability and computer-human interface design, and there was resistance from the users who had little or no experience with the technologies to be employed in the replacement. A "user-centered" design process evolved to address these issues. This paper presents the aspects of the process that had the greatest impact, and its effect on the resulting system.

2. USER-CENTERED DESIGN

The NOCC replacement involved 25 software engineers and support staff; development of over 200 K lines of code; 4 years duration; 26 computers connected to 4 Local Area Networks; and a significant amount of commercial off-the-shelf (COTS) software. Design decisions and technical trade-offs were made with a bias towards usability and operability. At the start of the project an approach was established to identify and address the key issues facing the developers. The following discussion describes the three steps in the process and examples of how they were applied.

Step #1 - Determine the design criteria for user concerns This required three considerations. First establish the area to be considered. Then specify how it might affect data representation. Finally consider how the presentation to the user should be made. For example:

How can the user interpret the data, given that NOCC receives several thousand status, configuration and performance parameters every five seconds from 15 antennae and associated ground equipment? By grouping information into a hierarchical structure and

providing algorithms to "bubble-up" interpretations, the amount of data that a user needs to work with at any one time is minimized. The presentation to the user is accomplished through navigation aids that traversed such a hierarchy.

How can an operator identify a problem and isolate its cause in a timely manner? The hierarchical structure of the data assists in this area. The presentation identifies a problem to the user by using color coding or flagging of problems and provides navigation aids that lead from the high-level aggregate view to detailed causes.

With operator turnover increasing how can the system aid in allowing use without in depth knowledge of DSN? Data representation has little impact in this area. This is in the purview of presentation. Consistent application of color coding and user interface simplifies training of new users.

Step #2 - Assess the impact of design decisions on the users Since the user community had little or no experience with this technology, getting them to participate was difficult. Some resistance early in the project was noticed. These areas were addressed through two steps. A formal Operability Design Document (ODD) was established that stated in user terms the criteria for system design. Then, during the entire development process a liberal use of prototypes was made. A DSN user interface prototype laboratory was established and experimentation with COTS software begun. This was all aided through collaboration with other JPL projects. For example:

The prototype of the hierarchical displays was widely demonstrated and led to a greater understanding of the idea of graphical user interfaces by those not yet exposed to the idea. Also acceptance of the hierarchical data representation and navigation aids was achieved.

Step #3 - Determine the technical aspects of the implementation (tools, platforms, etc.) Due to procurement requirements, the operating environment and hardware configuration were established earlier than would have been ideal. UNIX was specified as the operating system, and prototypes established an architecture for handling data rates and storage

capacities. However tools selection, changes to hardware to accommodate the user interface and other considerations evolved. For example:

Whereas DV-Draw tool was used during prototype development, other tools were used in final development. Demonstration to users established Motif with X-Windows as the presentation interface. Even with reduction of the data through hierarchical organization, a need evolved to display more data than was possible with a single monitor. A multiple monitor system was proposed even though it meant added difficulty to the developers. This yielded a twelve-fold increase in display area.

3. THE RESULTING NOCC SYSTEM

NOCC is a data acquisition system, configured as a network of two dozen identical Sun SparcStation 2 computers. Data acquisition and evaluation are performed with front-end computers connected to antennae around the world. These units broadcast to the back-end user workstations. Work consoles are equipped with triple screen monitors, mouse, printers and other needed facilities. Display data are refreshed every five seconds. Specialized computers handle the network management and file management functions. The system is fully redundant including the Local Area Networks. The NOCC hardware architecture is presented in Figure 1.

A graphical model of the DSN was constructed that represented the hierarchical structure of the data. The highest level display is a "checker board" where each square represents a DSN antenna-spacecraft pair. (Refer to the DSN Spaceraft Summary Display in Figure 2.) The next level shows a left-to-right data flow with assemblies represented by boxes connected by lines representing links. (Illustration is provided by the Telemetry Summary Display in Figure 3) The lowest level shows details of the assemblies and a performance history plot. A "bubble-up" algorithm allows proper representation of status at each level. (An assembly in a red or non-functional condition does not necessarily imply a red condition at the top level.)

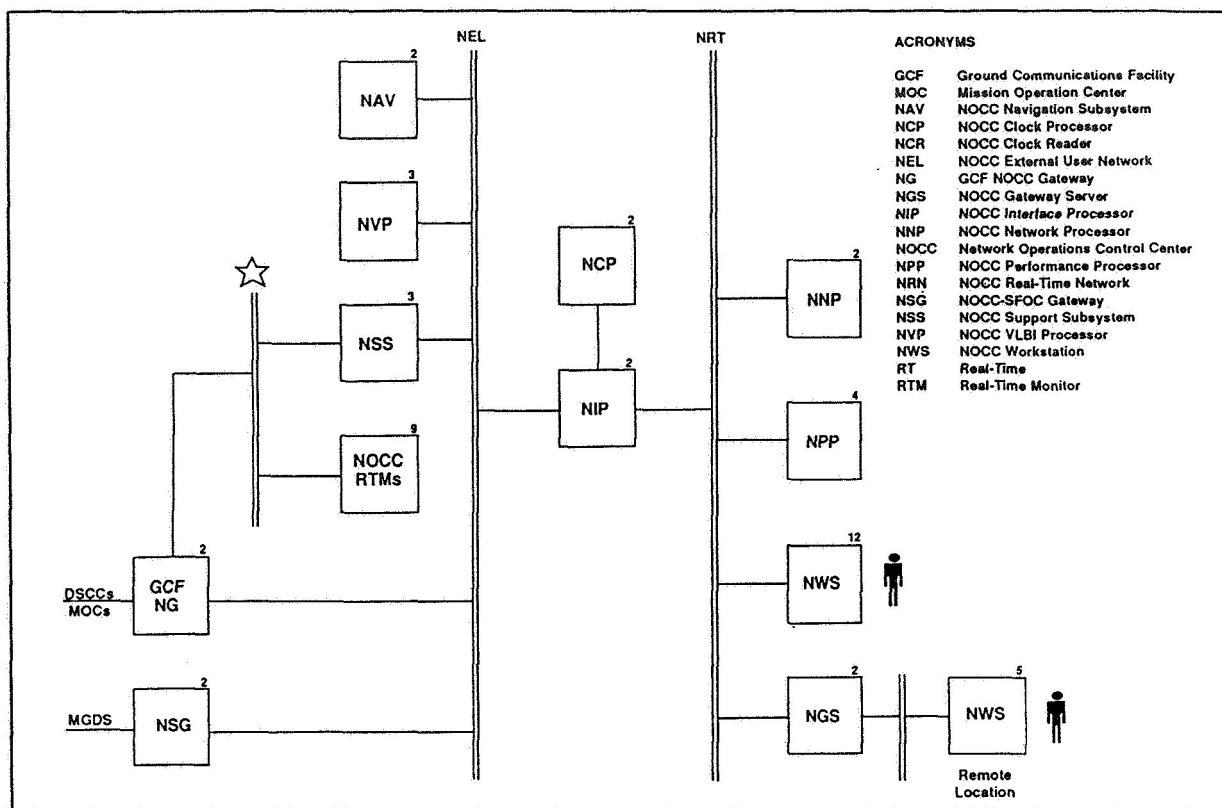


Figure 1. NOCC Hardware Architecture

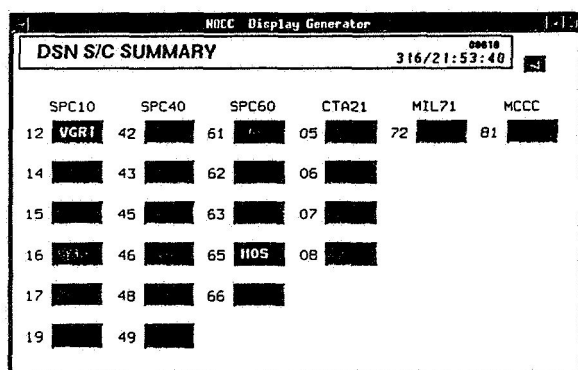


Figure 2.
DSN Spacecraft Summary Display

Navigation from top-to-bottom is accomplished by clicking the appropriate button for the element about which the user wishes more detail. Bottom-to-top and side-to-side navigation are accomplished with arrows.

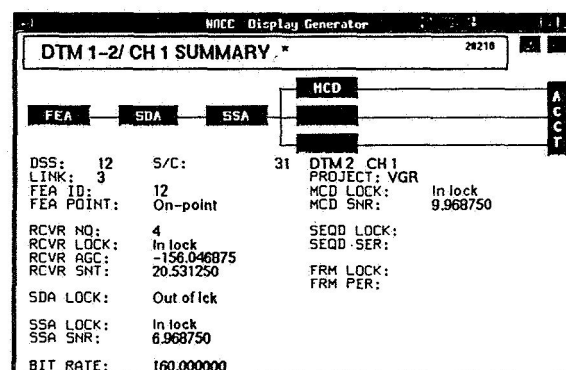


Figure 3.
Telemetry Summary Display

Color coding provides a visual means of representing status, performance and configuration. Green, yellow and red are used to denote normal operation, marginal conditions and critical alarms respectively. An unassigned link is highlighted with grey. Since more

than one lower level assembly can contribute to a condition, algorithms have been developed to relate the conditions. Algorithms are based on decision tables, Figure 4 explains the conditions and decisions for color coding the Antenna Summary button.

CONDITION	a	b	c	d	e
1. Data Validity valid invalid	Y				Y
2. Ant Operating Status out-of-service operational critical		Y		Y	Y
3. Ant Pointing Status on-point off-point			Y		Y
ACTION					
1. Set Antenna Summary Status = "blue"	X	X			
2. Set Antenna Summary Status = "green"					X
3. Set Antenna Summary Status = "yellow"					
4. Set Antenna Summary Status = "red"			X	X	

Figure 4.
Antenna Color Decision Table

The use of a **command line interface** and operator directives has been minimized through use of point and click, pop-up menus and pull-down menus.

An operator's workstation is a fully functional system providing support for a variety of tools. Along with the NOCC system are terminal emulation, screen or window dump/print, file transfer, word processing, spreadsheet, calculator and E-mail. Refer to Figure 5 for a specification of the NOCC Workstation.

Configuration tables and stored profiles control nearly all aspects of the system. A work station can be customized at login by use of stored profiles. Configuration tables specify items such as supported spacecraft and antennae, history and operator log entries, and what functions are authorized by user type.

Expandability can be accomplished through the addition of more workstations. Added functions or new user types can be supported on the same hardware with new profiles and custom programs. This type of expansion does not compromise usability or performance.

4. CONCLUSION

The user-centered approach not only led to a superior system, but also provided a high level of user satisfaction. NOCC has become a model for other development efforts.

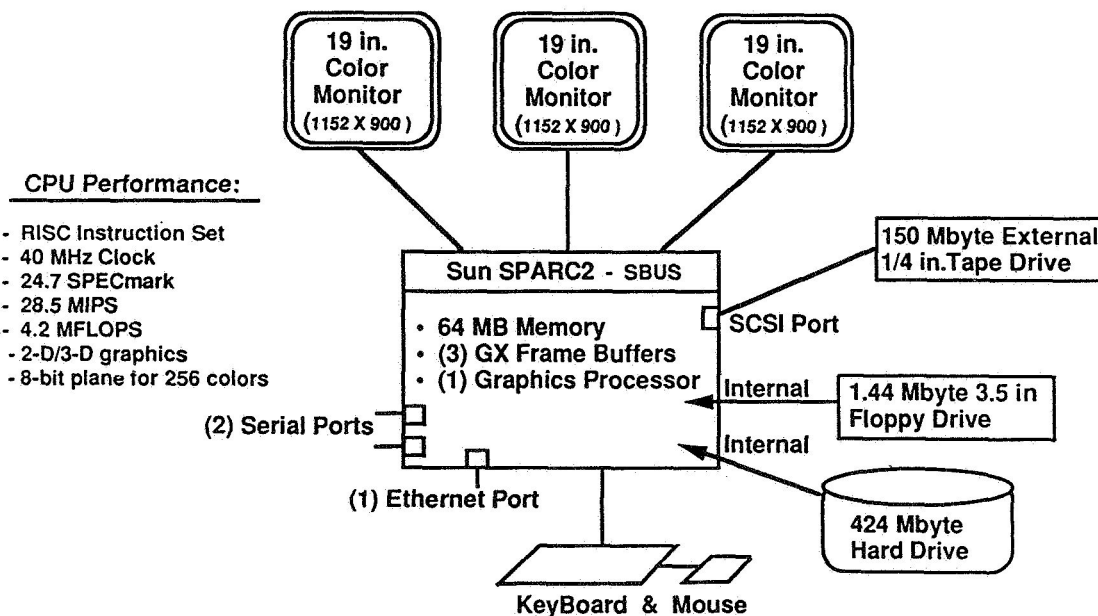


Figure 5. NOCC Workstation

N94-23918

A Method for Interference Mitigation in Space Communications Scheduling

Yen F. Wong

James L. Rash

NASA Goddard Space Flight Center, Greenbelt, Maryland

ABSTRACT

Increases in the number of user spacecraft and data rates supported by NASA's Tracking and Data Relay Satellite System (TDRSS) in the S and Ku bands could result in communications conflicts due to mutual interference. More attention must be paid to this problem in terms of communications scheduling. A method based on consideration of all relevant communications parameters has been developed to mitigate interference while minimizing unnecessary scheduling restrictions on both the TDRSS network and user resources. This method calculates required separation angles at TDRS and produces potential interference intervals, which can be used in the production of schedules free of unacceptable interference. The method also can be used as the basis for analysis, evaluation, and optimization of user schedules with respect to communications performance. This paper describes the method and its proposed application to scheduling in space communications. Test cases relative to missions operating at Ku-band, including Space Shuttle, are discussed.

Key Words: Interference mitigation, space communications, scheduling, S/I

1. INTRODUCTION

Scheduling of user spacecraft communications via the geosynchronous data relay satellites of NASA's Tracking and Data Relay Satellite System (TDRSS) must increasingly be concerned with the effects of mutual interference among users (Refs. 1-3). The ultimate objective is to schedule interference-free communications while minimizing constraints imposed on both users and network resources. This objective cannot be accomplished absent the capability to analyze, evaluate, and optimize user schedules with respect to communications performance. Consideration of the effect of communications factors such as signal to interference level ratio (S/I), bit error rate (BER) margin degradation, and power received is beyond the scope of current TDRSS network scheduling systems, giving rise to the need for a new generation of scheduling systems incorporating interference mitigation capabilities, especially in the late 1990's and beyond.

This has led to the development, within the Communications Link Analysis and Simulation

System (CLASS) at NASA Goddard Space Flight Center (GSFC), of a method for interference mitigation in space communications scheduling. CLASS is a software tool used for the prediction and evaluation of TDRSS/user spacecraft communications link performance. CLASS considers all communications channel parameters that affect link performance, including interference (Ref. 4).

The U. S. Air Force and the Jet Propulsion Laboratory have been using computer-generated predictions of interference as a scheduling constraint to avoid interference at a ground station receiving signals from two spacecraft and at two or more user spacecraft in the beam of a transmitting ground station (Ref. 5). Recently, NASA has begun evaluating predicted potential interference intervals, obtained as described below, as an adjunct to schedules generated by a partially-automated TDRSS network scheduling system; monitoring in the TDRSS network control center has revealed very few instances of unpredicted interference (Ref. 6).

The next section presents a TDRS telecommunications overview. Following sections describe a communications performance model; an approach to interference mitigation via required separation angles and potential interference intervals; and an interference mitigation procedure for scheduling. Then, illustrative test cases are presented, with discussion, followed by a summary.

2. TDRSS TELECOMMUNICATIONS OVERVIEW

NASA's TDRSS consists of a space segment and a ground segment. The ground segment of TDRSS consists of a ground terminal at White Sands, New Mexico. The operational space segment consists of a user transponder on each user spacecraft and three in-service satellites in geostationary orbit at 41, 171, and 174 degrees west longitude.

TDRSS provides telecommunications in S band via single access (SA) and multiple access (MA) services, and in Ku band via the SA service. Forward links (signals from the ground station via TDRS to a user) operate at data rates from 0.1 Kbps to 25 Mbps, and return links (signals from a user to the ground station via TDRS) operate at data rates from 0.1 Kbps to 300 Mbps (Ref. 7).

3. A MODEL FOR COMMUNICATIONS PERFORMANCE IN THE PRESENCE OF MUTUAL INTERFERENCE

In this paper, interference mitigation concerns only user return channels and single interferers. Multiple simultaneous interferers are not considered.

The proposed method to achieve interference mitigation uses BER margin degradation, formulated as a function of signal to interference level ratio (S/I), as the basic parameter for determination of channel communications performance for a link in the presence of interference (Ref. 8). The calculation of BER margin degradation includes all relevant parameters, such as PN or non-PN coding, frequency separation between desired user and interferer, channel coding, data rate (bandwidth) difference between desired user and interferer, and implementation loss.

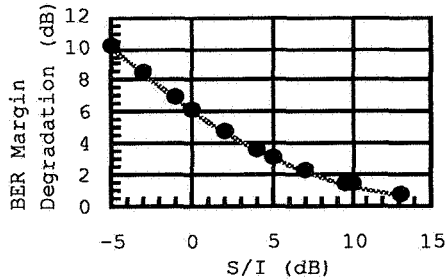


Fig. 1. Computed relationship between BER margin degradation and S/I. The desired user is Space Shuttle Orbiter using channel 3, 50 Mbps coded, and the interferer is operating at 50 Mbps (I+Q). Both are Ku band users. The desired user and interferer links are cross polarized with an assumed polarization rejection of the interfering signal of 15 dB on the TDRS SA antenna boresight.

Fig. 1 shows the relationship between BER degradation and S/I in a representative case. In general, degradation is computed by simulation, with S/I as an input parameter. Nonnegative BER margin is considered to correspond to acceptable communications performance when a link is degraded by interference.

The signal to interference level ratio S/I in dB at TDRS is defined as a function of the separation angle α between the desired user and the interferer as seen from TDRS:

$$\frac{S}{I}(\alpha) = (P_d + G(0)) - (P_i + G(\alpha) + R(\alpha)) \quad [\text{Eq. A}]$$

where P_d = the worst case (maximum range, given actual orbits) TDRS received power at unity antenna gain for the desired

user (in dB) including the loss due to the nonperfect polarization match between the TDRS and desired user antennas. It is assumed that the desired user is on the TDRS antenna boresight and that the desired user's antenna is pointing toward TDRS. P_d includes contributions from stochastic sources such as multipath (vehicle, earth, and atmospheric) and RFI.

P_i = the best case (minimum range, given actual orbits) TDRS received power at unity antenna gain for the interferer (in dB).

G = the TDRS antenna gain (in dB) as a function of the angle α . Spatially, G is assumed to be dependent only on the angle α .

R = the polarization rejection of the interferer's signal at the TDRS antenna (in dB) as a function of angle α . R always has a negative value when rejection is present (interferer oppositely polarized), and is zero otherwise. Spatially, R is assumed to be dependent only on the angle α .

Since only G and R are functions of α , Eq. (A) leads immediately to the following for any α_1 and α_2 , each an allowed value of α :

$$\frac{S}{I}(\alpha_2) - \frac{S}{I}(\alpha_1) = -[G(\alpha) + R(\alpha)]_{\alpha=\alpha_1}^{\alpha=\alpha_2} \quad [\text{Eq. B}]$$

From Eq. (B), S/I can be expressed as follows:

$$\begin{aligned} \frac{S}{I}(\alpha) = & -[G(\alpha) + R(\alpha)] \\ & + \frac{S}{I}(\alpha) + [G(0) + R(0)] \end{aligned} \quad [\text{Eq. C}]$$

which indicates that S/I, as a function of separation angle α , depends on the quantity $G(\alpha) + R(\alpha)$. This quantity, TDRS antenna gain plus polarization rejection, is referred to as the *adjusted TDRS antenna gain*.

CLASS makes available a simplified model for polarization rejection at the TDRS SA antenna, which is adequate for our purposes. In Fig. 2 the model is applied to an oppositely polarized user in Ku band (referred to as User A in the numerical example in Section 6), where it is assumed that the TDRS SA antenna boresight polarization rejection is 15 dB. Note that according to this model the polarization rejection is zero for all angles off boresight beyond

the first null of the antenna gain pattern (approximately 0.3 degrees in Ku band).

The TDRS SA antenna gain pattern envelope in Ku band, without polarization rejection adjustment, is shown in Fig. 3(a), representing the main beam, first null, and first sidelobe, with a logarithmic relation for the remainder of the pattern.

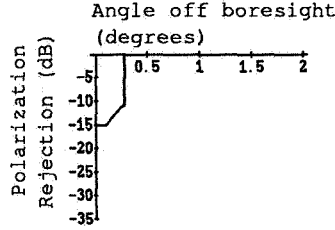


Fig. 2. Example of an assumed CLASS model of polarization rejection as a function of angle off boresight.

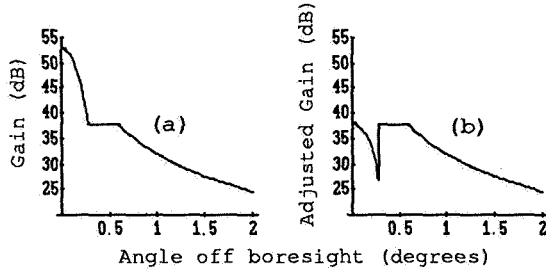


Fig. 3. TDRS SA antenna gain pattern envelope as modeled in CLASS: (a) without polarization rejection adjustment, and (b) adjusted by polarization rejection of a cross polarized signal (see Fig. 2).

Fig. 3(b) shows an example (in which the interferer is User A) of the adjusted antenna gain, $[G+R]$, in which the gain of the TDRS SA antenna is adjusted by the polarization rejection of an oppositely polarized user antenna. This figure results from adding the graphs shown in Figures 3(a) and 2.

With Eq. (C), the graph of S/I can be obtained from the negated adjusted TDRS antenna gain pattern, $-[G+R]$, shifted vertically by a constant (the boresight value of S/I plus the boresight value of $[G+R]$).

The negated TDRS antenna gain pattern envelopes for a representative case are shown in Figures 4(a) and 4(b), with and without polarization rejection, respectively. These graphs are the negation of the graphs in Fig. 3.

In general, due to the possibility of complex models for gain and polarization rejection, the negated adjusted antenna gain curve, $-[G+R]$, will have multiple relative minima. The global minimum

value of the curve may correspond to more than one value for the separation angle α (interferer's angle off boresight). We let α^* denote the least such value of α . Since, in general, the boresight value of $-[G+R]$ may be greater than the global minimum, α^* is not necessarily zero.

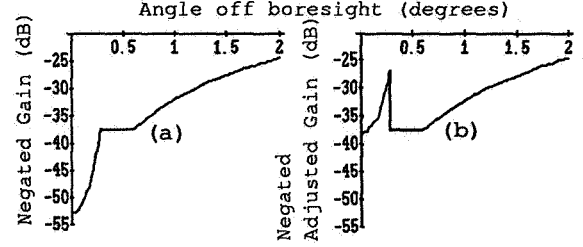


Fig. 4. Negated TDRS SA antenna gain pattern envelopes obtained as the negative of the graphs in Fig. 3. The global minimum of each curve occurs at boresight ($\alpha = 0$).

The graph of S/I is shown in Fig. 5 for representative cases (to be described later, in Table 4). By reference to the relationship between S/I and BER margin degradation (see Fig. 1), and with knowledge as to the maximum amount of BER degradation that can be tolerated by a given desired user signal, the S/I graph (Fig. 5) offers the means of finding a separation angle between the user spacecraft as seen from TDRS such that no unacceptable interference can occur. This is the basis of the method proposed in this paper for mutual interference mitigation.

3.1 Worst S/I

As the interferer moves off boresight, the interferer's received power changes in a manner dictated by the negated adjusted antenna gain curve (Fig. 4). When $-[G+R]$ reaches a local minimum, the interferer's power reaches a local maximum. Since the desired user remains on boresight, P_d remains constant. Therefore, as $-[G+R]$ reaches its global minimum (for example, at $\alpha = \alpha^*$), the value of S/I also reaches its global minimum. This minimum S/I is the "worst S/I ", and so we have, by Eq. (C), $(S/I)_{\text{worst}} = (S/I)(\alpha^*)$, and

$$\left(\frac{S}{I}\right)_{\text{worst}} = -[G(\alpha^*) + R(\alpha^*)] + \frac{S}{I}(0) + [G(0) + R(0)]$$

Note that if $\alpha^* = 0$, the worst S/I is simply boresight S/I : $(S/I)_{\text{worst}} = (S/I)(0)$.

3.2 Required S/I

Required S/I is defined as the value of S/I such that the degradation of the desired user signal equals the worst acceptable channel margin. The required S/I for any given combination of desired user and

interferer links is highly dependent on the specific signal structure, and is obtained by computer simulation. The worst S/I may or may not be less than the required S/I. If the worst S/I is less than the required S/I, then unacceptable mutual interference is possible for certain separation angles.

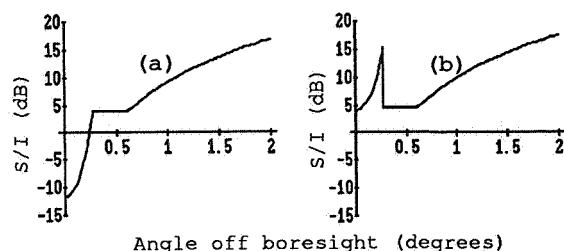


Fig. 5. S/I as a function of interferer's angle off boresight: (a) for a representative case where desired user and interferer have the same polarization but different signal levels, and (b) for the same case, except that the desired user and the interferer are oppositely polarized, showing the effect of TDRS antenna gain and polarization rejection on the interfering signal. Note that these curves have the shape of the negated TDRS antenna gain curve (unadjusted and adjusted, respectively (see Fig. 4)) shifted vertically by the value of the constant terms in Eq. (C).

4. INTERFERENCE MITIGATION VIA SEPARATION ANGLE AND POTENTIAL INTERFERENCE INTERVALS

4.1 Required separation angle

Since the desired user is assumed to be on the TDRS antenna boresight, and since antenna gain decreases off boresight, a sufficient variation of the interferer's separation angle provides discrimination between the signals, reduces the interference level, and increases the S/I level ratio. In the case where the worst S/I is less than the required S/I (that is, where interference is possible), the required S/I corresponds to specific separation angles that can be read directly from the graph of S/I (see Fig. 5). Note that, due to the possibility of multiple lobes in the adjusted antenna gain graph (and therefore multiple lobes in the graph of S/I), there may be multiple disjoint ranges of separation angle providing at least the necessary value of S/I. The largest of all the angles where S/I is equal to the required S/I is defined as the *required separation angle*. For purposes of user spacecraft communications scheduling, it must be assumed that any smaller separation angle will correspond to an unacceptable level of interference.

4.2 Potential interference intervals

A *potential interference interval* is defined as any time interval during which the separation angle between the two user spacecraft is less than the required separation angle as described above. During such intervals, unacceptable interference could occur if the given pair of links of the two spacecraft are active via the same TDRS. The potential interference intervals, therefore, would constrain any interference mitigation scheduling process by specifying when the two links should not be used simultaneously for communications to a given TDRS. A method to determine which of the two links should not be scheduled during any potential interference interval is part of the algorithm used by the scheduler and is beyond the scope of this paper.

Calculation of potential interference intervals is straight forward, given user orbits and required separation angles.

5. A PROCEDURE FOR INTERFERENCE MITIGATION IN SCHEDULING

A procedure is suggested for producing schedules free of unacceptable interference while minimizing restrictions on use of network and user resources. This procedure is based on a model, described above, for communications performance in the presence of interference, which permits calculation of required separation angles, by which potential interference intervals are obtained. The procedure is summarized as follows:

(1) For every pair of desired and interfering signals, determine $-[G+R]$ as a function of α (the separation angle at a given TDRS between the desired user and the interferer), where G is the TDRS antenna gain envelope and R is the polarization rejection of the interfering signal at the TDRS antenna. R is assumed to be zero if the desired user and interferer have the same polarization.

(2) For every pair of desired and interfering signals, determine the smallest separation angle, α^* , that corresponds to the global minimum value of the function $-[G+R]$.

(3) For every pair of desired and interfering signals, determine S/I, the signal to interference level ratio, as a function of α , given by Eq. (A) above. Calculate $(S/I)(\alpha^*) = -[G+R](\alpha^*) + (S/I)(0) + [G+R](0)$. This is the *worst S/I*.

(4) For every pair of desired and interfering signals, determine by computer simulation the degradation of the desired signal that corresponds to the worst S/I determined in step (3). This is the desired signal's *worst degradation*. Identify all signal pairs where the desired signal's worst degradation exceeds the desired user's worst acceptable channel margin. Unacceptable mutual interference is possible for these signal pairs.

(5) For every pair of desired and interfering signals where interference is potentially unacceptable as determined in step (4), determine by computer simulation the *required S/I*, that is, the S/I for which the degradation is equal to the desired signal's worst acceptable channel margin.

(6) For every pair of desired and interfering signals where interference is potentially unacceptable as found in step (4), calculate, from the S/I graph, the required separation angle (the largest separation angle between desired user and interferer that provides the required S/I as determined in step (5)).

(7) For every pair of desired and interfering signals where interference is unacceptable as determined in step (4), and on the basis of the separation angles obtained in step (6), find all *potential interference intervals*, that is, intervals during which unacceptable interference is possible. These calculations involve the actual orbits of the given spacecraft and the TDRSs.

(8) Use the potential interference intervals from step (7) as a constraint to a scheduler for generating schedules free of unacceptable interference. The effect of this constraint is to preclude the scheduling of at least one member of any pair of desired/interferer links via a given TDRS during any potential interference interval associated with that pair of links.

The first four steps can be used as a screening process to isolate the cases where unacceptable interference could occur. Steps (5) and (6) would be applied in all such cases, as an intermediate process prior to execution of a scheduler. Step (7) would be performed prior to every run of an interference mitigation scheduling system (step (8)).

6. APPLICATION OF THE METHOD: A NUMERICAL EXAMPLE

The proposed method has been applied to study interference between the Space Shuttle Orbiter (SSO) and TDRSS users operating at high data rates in Ku band.

All the users in this example operate with a carrier frequency of 15.0034 GHz, unspread.

Table 1 presents the Shuttle link parameters. SSO operates with Right Circular Polarization (RCP). Channels 1 and 2 are rate 1/2 convolutional coded and channel 3 is uncoded (Ref. 9).

User A link parameters are shown in Table 2. User A operates with Left Circular Polarization (LCP) at data rates of 300 Mbps and 50 Mbps. Table 3 presents the link characteristics for User B operating with RCP at a data rate of 300 Mbps.

The illustrative example discussed below considers two cases. In each case, the Shuttle, as the desired user, suffers from interference.

6.1 Interference analysis results

Table 4 presents the results of an example interference analysis for two cases.

In Case 1, the SSO (COLUMBIA) channel 3 (50 Mbps) experiences interference from User B's 300 Mbps link (I + Q). The worst S/I is less than the required S/I by 17.8 dB. The required separation angle can be obtained directly from the appropriate S/I graph (Fig. 5(a)): for the required S/I of 6.2 dB, the required separation angle is 0.74 degrees. There is no unacceptable interference in Case 1 for SSO channels 1 and 2.

In Case 2, where the interferer is the User A 50 Mbps link (I + Q), the worst S/I (4.0 dB) is less than the required S/I (9.0 dB) by 5.0 dB, and from Fig. 5(b) the required separation angle is seen to be 0.92 degrees. A comparison of Case 2 with Case 1 might seem to present an apparent inconsistency: as the interferer EIRP is nearly the same in both cases, and as the desired user in Case 2 is cross polarized relative to the interferer, it would seem reasonable to conclude that the required separation angle should be smaller than in Case 1. However, in Case 2, the interfering signal is "in-band", so that all of the interferer's power affects the desired channel, making the required S/I greater than that in Case 1. Also (see Fig. 2), it is clear why the advantage of cross polarization does not apply in Case 2, since the largest separation angle corresponding to the required S/I (see Fig. 5) is beyond the first null of the TDRS SA antenna. No unacceptable interference exists in Case 2 for SSO channels 1 and 2.

Unacceptable interference in Case 2 does not occur between the User A 300 Mbps link (I + Q) and SSO channels 1, 2, or 3.

6.2 Potential Interference Intervals

Potential interference intervals depend on the choice of orbits for user spacecraft. Fig. 6 illustrates this dependency by showing the intervals for two choices for the user orbital elements. The only difference between these choices is the value for the mean anomaly. For the choice illustrated in Fig. 6(a), the difference in the mean anomaly is zero degrees; for the choice illustrated in Fig. 6(b), it is 20 degrees. The total of the potential interference intervals goes from 100% of the in-view time (Fig. 6(a))—approximately 813 minutes during the 24 hour scheduling period—to 7.5% of the in-view time (Fig. 6(b))—approximately 61 minutes. Thus, the potential interference intervals become shorter and less numerous as the orbital spacing of the users increases. If, indeed, the mean anomalies differ by more than approximately 48 degrees, with all other factors in this example remaining the same, unacceptable interference becomes impossible, and potential interference intervals no longer exist.

7. SUMMARY

It is proposed that an interference mitigation scheduling system must reflect consideration of communications performance, and it is argued that the

use of BER degradation as a function of S/I is a sufficient basis for an interference mitigation scheduling system.

In general, scheduling may involve any number of user spacecraft. The scope of the approach presented in this paper is limited to the case of single interferers.

This paper presents a model of communications performance as affected by the presence of mutual interference. The model formulates communications performance in terms of S/I, which is considered as a function of the interferer's angle off the receiving antenna boresight. Required separation angles for interference mitigation can be calculated based on this

functional relationship, and these angles then can be used to determine potential interference intervals.

Potential interference intervals are proposed for use as a constraint in an interference mitigation scheduling system. By guaranteeing acceptable BER degradation for all desired user/interferer link combinations, except during potential interference intervals associated with those link combinations, the proposed procedure guarantees schedules to be free of unacceptable mutual interference. Potential interference intervals also can be useful as the basis for evaluating and optimizing (with respect to communications performance) user schedules produced by any scheduling system.

Table 1. Space Shuttle Orbiter Link Characteristics

CHANNEL	DATA RATE (Kbps)	EIRP (dBW)	LINK MARGIN (dB)
Channel 1: Subcarrier Q	192	39.4	19.0
Channel 2: Subcarrier I	2,000	43.6	13.5
Channel 3: Baseband	50,000	51.0	1.5

Table 2. User A Link Characteristics

CHANNEL	DATA RATE (Mbps)	EIRP (dBW)	LINK MARGIN (dB)
I	150	57.1*	3.0
Q	150	57.1*	3.0
I	25	57.1*	10.8
Q	25	57.1*	10.8

Table 3. User B Link Characteristics

CHANNEL	DATA RATE (Mbps)	EIRP (dBW)	LINK MARGIN (dB)
I	150	57.7**	3.6
Q	150	57.7**	3.6

* The minimum EIRP required to achieve a 3 dB margin for a 300 Mbps user.

** The minimum EIRP required to achieve a 3.6 dB margin for a 300 Mbps user.

Table 4. Analysis results for channels with unacceptable interference.

		Case 1	Case 2
Desired User	User ID	COLUMBIA	COLUMBIA
	Channel	3	3
	Polarization	RCP	RCP
	Worst Acceptable Channel Margin	1.5	1.5
Interferer	User ID	User B	User A
	Polarization	RCP	LCP
S/I	Required (dB)	6.2*	9.0*
	Worst (dB)	-11.6	4.0
Worst Degradation (dB)		**	3.5*
Required Separation Angle (deg.)		0.74	0.92

* Obtained by computer simulation.

** Degradation is much greater than 1.5 dB.

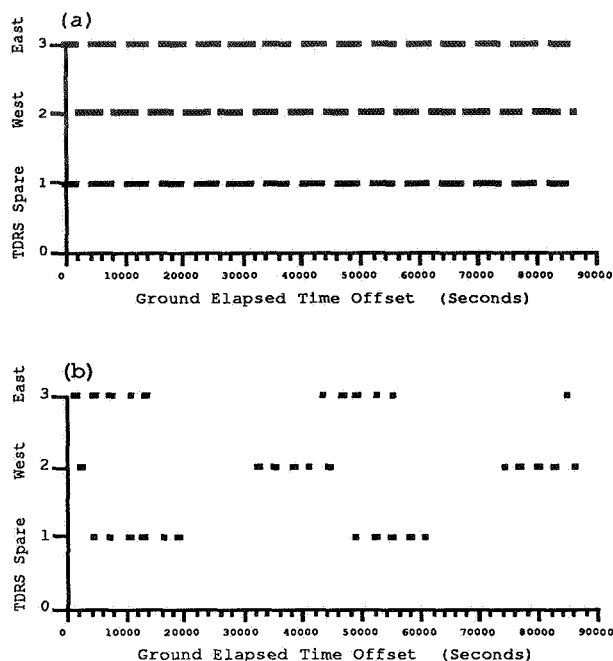


Fig. 6. Potential Interference Intervals at (1) TDRS Spare, (2) TDRS West, and (3) TDRS East when the desired user is SSO COLUMBIA and the interferer is User A. A period of 24 hours is represented. Each user spacecraft orbit is approximately 90 minutes in duration. (a) The users have identical orbits, so that the separation angle is always zero degrees during TDRS view periods. (b) The users have identical orbits except for a difference of 20 degrees in their mean anomalies. In each orbit there are two times when they are separated by less than the required separation angle: once just after they both appear above the horizon as seen by the TDRS, and once just before they both disappear below the horizon.

8. ACKNOWLEDGMENTS

The authors would like to acknowledge CLASS Project Manager Robert Godfrey of NASA/GSFC for originating the concept of an interference mitigation scheduling system for CLASS. We would also like to thank Nancy Smith and Ron Vento of NASA/GSFC, Farzad Ghazvinian and Masoud Toufanian of LinCom Corporation, and David Wampler of Stanford Telecommunications for valuable technical support. Finally, we would like to express appreciation to our supervisor, Frank Stocklin, for supporting our efforts on this project.

9. REFERENCES

1. NASA Goddard Space Flight Center. December, 1990. Space Network Mission Model and Network Loading (Update, March 1992). Presentation by A. Levine, GSFC/Code 534.2, to NASA management. Greenbelt, Maryland.
2. Computer Sciences Corporation. February, 1990. *Network Control Center User Planning System System Requirements Document*, 514-4SRD/0389. Greenbelt, Maryland.
3. Welti, G., Christopher, P., and Griffin, M. March, 1990. *ATDRSS Self Interference Assessment*, Technical Report TR0007, Stanford Telecommunications, Inc. Reston, Virginia.
4. NASA Goddard Space Flight Center. September, 1989. *Communication Link Analysis and Simulation System*, CLASS No. 101. Greenbelt, Maryland.
5. Sollfrey, W. October 21-24, 1984. Interference problems for nongeostationary satellites. In *Conference Record, Military Communications Conference (Milcom '84)*, McLean, Virginia, Vol. 3, pp. 384-388.
6. Wampler, D. and Kaplan, T. March 23, 1992. *ACRS Performance Evaluation*, Technical Report 028-1DW92, Stanford Telecommunications, Inc. Reston, Virginia.
7. NASA Goddard Space Flight Center (1988). *Space Network (SN) Users' Guide*, STDN No. 101.2, Revision 6. Greenbelt, Maryland.
8. Bhargava, V. K., Haccoun, D., Matyas, R., and Nuspl, P. P. 1981. *Digital Communications by Satellite*. New York, NY: John Wiley & Sons.
9. NASA Goddard Space Flight Center. January 30, 1989. Memo, To: Distribution, From: GSFC Code 531.1, Subject: STS Link Calculations. Greenbelt, Maryland.

SESSION H

NAVIGATION

CO-CHAIRPERSONS

J. K. Campbell, JPL, California Institute of Technology, USA
J. P. Carrou, CNES Toulouse Space Centre, France
J. P. McDanell, JPL, California Institute of Technology, USA

Summary 575
J. P. McDanell

H.1 Operational Navigation Support for the TOPEX/Poseidon Mission 577
R. B. Frauenholz, JPL, California Institute of Technology, Pasadena, California, USA

H.2 Real-Time On-Board Orbit Determination with DORIS 579
J.-P. Berthias, C. Jayles, and D. Pradines, CNES Toulouse Space Centre, France

H.3 TOPEX/Poseidon Precision Orbit Determination Production and Expert System 585
B. Putney, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA;
N. Zelensky and S. Klosko, Hughes STX, Lanham, Maryland, USA

H.4 A Real-Time Navigation Monitoring Expert System for the Space Shuttle Mission Control Center 591
L. Wang and M. Fletcher, NASA Johnson Space Center, Houston, Texas, USA

H.5 XRTD: An X-Windows Based, Real-Time Radiometric Display and Analysis System 601
V. M. Pollmeier, JPL, California Institute of Technology, Pasadena, California, USA

H.6 Kalman Filtering Applied to Real-Time Monitoring of Apogee Maneuvers 607
F. de Boer, Matra Marconi Space, Toulouse, France; C. Barbier, Alcatel Espace, Toulouse, France

H.7 Mars 94/96: The French Navigation Tasks 613
F. Bonneau and J. Bernard, CNES Toulouse Space Centre, France; D. Delobette, Sema-Group, Toulouse, France

H.8 MERCATOR: Methods and Realization for Control of the Attitude and the Orbit of Spacecraft 619
G. Tavernier and G. Campan, CNES Toulouse Space Centre, France

H.9 Navigation of Space VLBI Missions: Radioastron and VSOP 625
J. Ellis, JPL, California Institute of Technology, Pasadena, California, USA

**H.10 The Geo-Control System for Station Keeping and Colocation
of Geostationary Satellites 629**

O. Montenbruck and M. C. Eckstein, DLR German Space Operations Centre,
Oberpfaffenhofen, Germany; J. Gonner, Société Européenne des Satellites, Betzdorf,
Luxembourg

**H.11 HISPASAT Launch and Early Operations Phases: Computation
and Monitoring of Geostationary Satellite Positioning 635**

P. Brousse, CNES Toulouse Space Centre, France; A. Desprairies, Matra Marconi
Space, Toulouse, France

SUMMARY — SESSION H

J. P. MCDANELL
Jet Propulsion Laboratory

This session consisted of 11 papers. The session was divided into two parts, with four papers in the Wednesday afternoon segment and seven on Thursday morning. All scheduled papers were presented. The distribution of papers among agencies was somewhat skewed, with five from CNES, three from JPL, and one each from the German Space Operations Centre (GSOC), NASA Goddard Space Flight Center, and NASA Johnson Space Center. This distribution was due in part to the need for the chairpersons to recruit additional papers to complete the session, since there was a shortage of responses to the initial call for papers.

Attendance was very good, especially considering the specialized subject matter. Attendance ranged from 50 to 60 in the Wednesday session and from 40 to 50 on Thursday. Based on comments received afterward, the general reaction to the session was quite positive. However, there were relatively few questions from the audience, which was somewhat disappointing.

The navigation session included two demonstrations, which made use of a SUN workstation, scan converter, and video projector to provide a large-screen image of the monitor display while the speakers went through their presentations at the keyboard. The second demonstration could not be completed because of a software problem, but interested parties were later able to view it in the demonstration area at the Pasadena Center.

The variety of topics covered in the session is summarized as follows (some of the papers addressed more than one topic):

- Navigation operations experience – 3 papers
- Future mission plans – 2 papers
- Software system designs – 4 papers
- Expert systems applications – 2 papers
- Graphics applications – 3 papers
- Real-time monitoring – 3 papers
- Onboard orbit determination – 1 paper
- Demonstrations – 2 papers

The papers reflected some important trends in navigation operations. One such trend is a shift of focus in software and system design to emphasize easy-to-use, information-rich graphical displays of results. The mathematical models and algorithms that were the dominant focus of navigation design in years past have reached a relatively high state of maturity. This factor — and the availability of enabling technology — have led to greater emphasis on operability, reliability, and improved analysis tools to aid in the interpretation of results.

A second trend — related to the first — is the infusion of technology to enable rapid solution of large-scale problems using data sets from many sources. As a result, complex navigation problems that might have been considered infeasible only a few years ago are being solved routinely. Further evidence of technology infusion is the fact that relative newcomers to space navigation have been able to develop impressive capabilities quickly with a modest investment of resources.

By most criteria, the session was very successful. It was characterized by high-quality presentations of varied topics that represented important trends in the discipline area. Navigation practitioners from different agencies with experience in a variety of applications described their respective challenges and exchanged ideas on tools, techniques, and systems for meeting those challenges.

OPERATIONAL NAVIGATION SUPPORT FOR THE TOPEX/POSEIDON MISSION

R. B. Frauenholz
JPL - California Institute of Technology
Pasadena, CA

*Presented but not submitted to proceedings
as of December 14, 1992.*

Real-time on-board orbit determination with DORIS

N 9 4 - 2 3 9 1 9

J.-P. Berthias, C. Jayles, D. Pradines

Centre National d'Etudes Spatiales
18 Av. Edouard Belin
31055 TOULOUSE CEDEX
FRANCE

Abstract: A space borne orbit determination system is being developed by the French Space Agency (CNES) for the SPOT 4 satellite. It processes DORIS measurements to produce an orbit with an accuracy of about 50 meters rms. In order to evaluate the reliability of the software, it was combined with the MERCATOR man/machine interface and used to process the TOPEX/Poseidon DORIS data in near real time during the validation phase of the instrument, at JPL and at CNES. This paper gives an overview of the orbit determination system and presents the results of the TOPEX/Poseidon experiment.

Introduction:

Autonomous control of a satellite's orbit is an attractive concept which offers many advantages. Under the control of an on-board program, the optimum firing of the thrusters maintains the spacecraft on an orbit which has been pre-programmed. Meanwhile on-board instruments use the knowledge of the position and velocity of the spacecraft to orient themselves or pre-process their measurements. On the ground, operators at a simplified control center use telemetry data to verify the proper behavior of the spacecraft.

Partial autonomy is already achieved by all satellites. Attitude control is a nearly completely automated task which is under the control of an on-board program. Since orientation measurements are performed by the spacecraft itself, and a short response time is necessary, the automation of attitude control is not only convenient but practically indispensable. On the other hand, until the last few years, the data necessary for precise navigation could only be obtained on the ground and required lengthy processing. Thanks to advances in the stability of on-board clocks, new on-board tracking systems including GPS and DORIS are now available. With these instruments the accurate measurements needed to compute the position and velocity of the spacecraft are available on-board, thus opening the door to automated navigation.

As a first step in this direction, the French Space Agency, CNES (Centre National d'Etudes Spatiales) decided to pursue the subject of on-board orbit

determination. First, a feasibility study was conducted in collaboration with Aérospatiale and Dassault Electronique. It demonstrated the possibility of computing the orbit of a low altitude satellite equipped with a DORIS receiver, using only a small program and limited computing resources. Following this success, it was decided to fly an on-board orbit determination system on the Earth observation satellite SPOT 4, and the DIODE project was thus created. Some details of this system will be given in this paper.

The presence of a DORIS receiver on TOPEX/Poseidon provided us with an opportunity to test the current version of the software in a new environment and validate some of our assumptions. In return, it offered a quick-look approach to the TOPEX/Poseidon DORIS orbit and a immediate appraisal of the performance of the instrument.

DORIS:

The DORIS system, developed jointly by CNES, the French National Geographic Institute, IGN (Institut Géographique National) and the GRGS (Groupe de Recherches en Géodésie Spatiale), uses a global network of ground beacons which broadcast omnidirectionally on two frequencies, 2036.25 and 401.25 MHz. Each beacon contains an ultrastable quartz oscillator with a stability of about $5 \cdot 10^{-13}$ for periods of 10 to 100 seconds. In addition, two master beacons, one in Toulouse and one in Kourou, are tied to atomic clocks. The DORIS instrument carried by

the spacecraft contains a dual frequency receiver and an ultrastable oscillator. It computes the Doppler shifts and the time of arrival of the time synchronization signals. The overall system noise is lower than 0.3 mm/s [1].

The instrument computes the Doppler count at both frequencies every 10 seconds when a beacon is in view. In nominal operation mode, the receiver is programmed to listen to specific beacons, one beacon at a time.

The orbital coverage provided by the extensive beacon network is over 70 % for the SPOT satellites (800 km altitude) and reaches 85 % for TOPEX/Poseidon (1330 km altitude).

Implementation:

To accommodate the on-board orbit determination software, Dassault Electronique, which manufactures the DORIS receiver, has added a new card to the instrument. This new card carries a Marconi MAS 281 microprocessor, which follows the MIL STD 1750A standard, and 48 kwords of available memory. The clock rate is 10 MHz. EDACs are used to detect and eliminate memory upsets.

The memory of the new card is seen by the DORIS receiver as part of its memory. This allows for easy loading of the software while in flight, and permits the use of reserved memory locations for an interface between the receiver and the orbit determination system (cf Fig. 1).

The orbit determination software is being developed by CNES. It runs under the supervision of the navigation management software written by Dassault Electronique, which is responsible, among other things, for the interrupt driven interface with SPOT 4.

Both sets of software are written in Ada. The orbit determination program is designed following the Hierarchical Object Oriented Design (HOOD) method. With the help of the CNES Ada group, STOOD, a HOOD implementation tool developed by TNI, has been used to design and write the software.

The result is a 2400 line long code which is easy to maintain and update. When compiled for the 1750 target microprocessor using the VAX based TLD cross-compiler, the code and data occupies about 32 kwords.

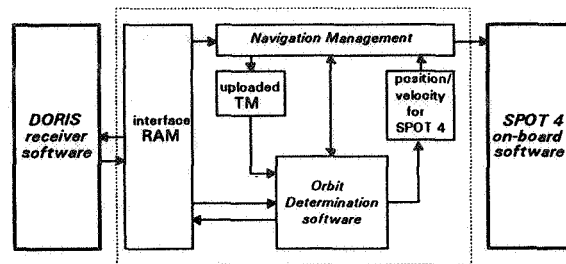


Fig. 1 - System architecture

Principle of operation:

Once an initial state vector has been uploaded, the function of the on-board orbit determination program is to update it every ten seconds. As the DORIS measurements are performed on-board, they are dated in instrument time, and a correspondence between instrument time and TAI has to be maintained to provide properly dated state vectors.

Two separate Kalman filters are used to fulfil these functions. The first filter computes the difference between instrument time and TAI. This value, modeled as a constant, is updated when synchronization measurements are performed while flying over a master beacon. The second filter updates a state composed of the spacecraft position and velocity and a frequency bias for the current pass when measurements are performed. This filter is written using Bierman's UD formulation.

The tuning parameters for both filters can be modified by ground commands.

When the navigation management software activates the orbit determination function, it provides the date at which the state is to be computed and the list of uploaded parameters to process. The date is equal to the TAI time of the end of the preceding cycle, rounded to an even multiple of 10 seconds.

After activation of the orbit determination function by the navigation management software, the schedule of operations is as follows:

- 1) If uploaded parameters are present, they are taken into account, and the values of the corresponding parameters are modified.
- 2) The current state is extrapolated to the desired date. The extrapolation is performed by a fourth order Runge-Kutta integrator using the space efficient Gill formulation.

3) If measurements are performed over a master beacon, the difference between instrument time and TAI is updated.

4) If measurements are performed, the predicted value and the residual are computed. If the residual is lower than the acceptance threshold, the measurement is used to update the state.

All these operations are performed in less than five seconds.

If for some reason the orbit navigation function has not been activated over a few cycles, at the next call the difference between the date at which the state should be computed and the date of the current state can be large. As a maximum of five steps can be performed per call, it can take many calls for the system to catch up. During this interval, no measurement is processed and no result is provided to the user.

Models:

DORIS, a Doppler based tracking system, measures relative velocities. To relate the measured data to the spacecraft position, an integration of the equations of motion is required. Hence, a force model is necessary.

Due to the excellent orbital coverage, periods during which the extrapolated position is not corrected by a measurement are very short. Therefore errors due to limitations in the force model cannot grow large. The current force model is limited to a dedicated sixth order and degree gravity field provided by the GRGS. This choice satisfies the accuracy requirements and reduces the memory requirement to a minimum. Third body perturbation and drag models have been developed, and could be easily included in the software if the mission justified it and if space was available.

As the acceleration model is rudimentary, the model covariance input to the filter is large. This in turn gives a lot of weight to the measurements. The orbit is data driven, and this can be useful when an unexpected force is present. Least squares based standard orbit determination software cannot handle unmodeled forces, while this system can. In this respect, this system produces results similar to those of a partly dynamic and partly geometric analysis.

Measurement models are also very simple. Dual frequency ionosphere correction is applied to the data. A basic troposphere delay correction, assuming a constant vertical electron content, is also applied. The

Earth orientation model is almost non-existent: no correction is applied for precession, nutation, or polar motion. The Earth rotation rate is assumed constant and UTC is used in place of UT1 to compute the hour angle. Beacon locations are memorized in an on-board table which can be updated by ground commands.

The choice of these models has been validated by processing many samples of SPOT 2 DORIS data. In particular, studies proved that the accuracy requirements are met even in the presence of the increased drag due to high solar activity or of frequent thruster firing such as periods when the spacecraft goes into safhold mode [2].

The TOPEX validation experiment:

The launch of TOPEX/Poseidon on 10 August 1992 provided an excellent opportunity to test the behavior of the orbit determination component of the system. Processing data from a new instrument is useful to ensure that no assumptions were made that are unique to the SPOT 2 configuration.

Also, the high level of spacecraft activity between the launch and the transfer into nominal orbit requires the transmission to the software of many parameters, such as maneuver characteristics.

It was therefore decided to run the software in near real time to process the TOPEX/Poseidon DORIS data both at JPL and at CNES. The environment for the orbit determination program was provided by MERCATOR.

MERCATOR is regularly used by CNES to put geostationary satellites in orbit [3]. It runs on SUN workstations and offers a framework within which technical modules can easily be included and activated. It was successfully employed for ten missions between April 1989 and September 1992. It is therefore a solid and reliable software which is ergonomically adapted to the needs of orbital mechanic engineers.

For the TOPEX/Poseidon experiment MERCATOR was used to

- decommutate the telemetry and supply the data in a form similar to that coming out of the DORIS instrument
- provide a user friendly interface
- permit the rapid analysis of results using auxiliary calculations and graphical tools

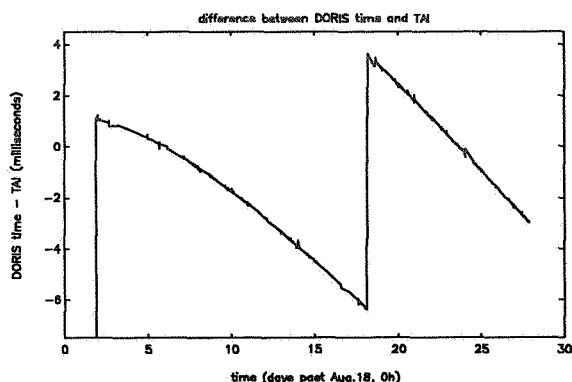


Fig. 2 - Time synchronization

Telemetry from the DORIS instrument on the TOPEX/Poseidon spacecraft is recorded on-board and then played back to Earth via TDRSS. From this data the TOPEX/Poseidon Ground Segment generates a Selected Telemetry Record (STR) file daily. These files are then copied to the CNES control center via an electronic network.

For our experiment, we could access these files both at JPL and at CNES. After the first few days, once a file was copied to our SUN, it took about half an hour to produce an ephemeris.

In addition to this regular operation mode, we also processed special request STRs at JPL, using initial state vectors provided by the TOPEX/Poseidon navigation team. We were then able to produce short arc ephemerides only one to two hours after the acquisition of the data. This was especially helpful to check the state of the instrument after major operations (reset of the clock, change in the mode of operations, etc.).

Overall, the quality of the results exceeded our expectations. In one instance, we had to integrate over an eight hour data gap, and were amazed to see the filter locked back on the right orbit on the first subsequent pass. Note that this would probably not be possible for the lower altitude SPOT spacecraft.

In order to more accurately reproduce the conditions of on-board operations, we have reprocessed in a single batch the first 28 days of DORIS data following the second reset of the instrument clock. The uploaded parameters were the initial conditions, the date of the reset of the clock, the value of UTC minus TAI, and the pre-execution values of the five maneuvers (CAL, INC, IPM1, IPM2, IPM3) [4]. With this limited set of inputs, the orbit determination software produced the results shown below.

Figure 2 shows the difference between DORIS time and TAI as computed by our orbit determination software using synchronization measurements performed over the Toulouse master beacon. The curve clearly shows the clock adjustment of Aug. 19 followed by the slow drift of the DORIS time. Note that at first the oscillator drifts, but, as the oscillator reaches a stable state, the difference becomes linear, due to the fact that the oscillator frequency is not exactly nominal. On the 18th day, an automated clock correction brought the DORIS clock back to TAI. The error level in the time determination is at the level of a few tenths of milliseconds. This corresponds to a meter size error on the orbit in the along-track direction. Therefore, the time synchronization is not a limiting factor on the orbital accuracy.

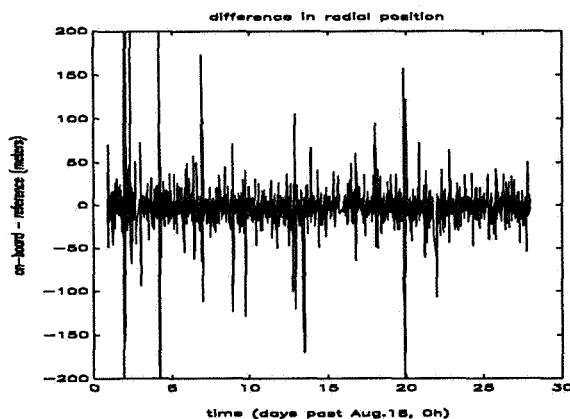


Fig 3 - Radial difference

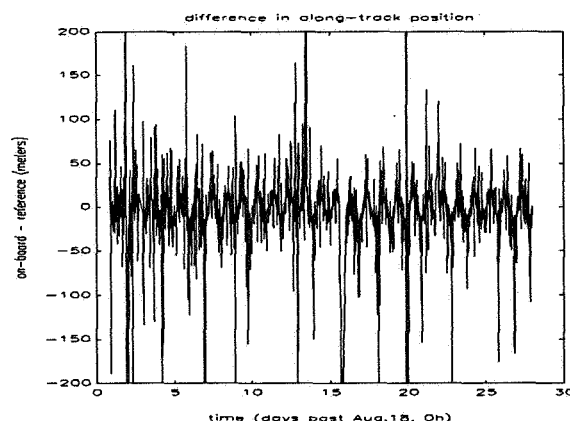


Fig 4 - Along-track difference

Figures 3 and 4 show differences between the "on-board" orbit and a reference orbit obtained by processing the DORIS data with the ZOOM software [5]. The precision level of the reference is better than one meter.

The standard deviations corresponding to figures 3 and 4 are 20 and 44 meters respectively. The overall standard deviation of the distance to the reference orbit is 53 meters. These results are therefore within the requirements of the SPOT 4 project.

However, a few accidents contributed greatly to increase these statistics. In particular, this test showed us that our software does not properly handle forced clock resynchronizations, such as the one which happened on August 19. As these occur only after a restart of the instrument, we had never previously experienced one of those in our tests.

It should be noted that maneuvers do not appear on these curves, except when there is a data gap, and therefore a reference orbit gap. On the 15th day of the arc, the apparent dip in the along-track curve is due to a lack of reference. This is repeated in the radial curve on the 22nd day.

A periodic term with a daily period and an amplitude of about 20 meters can be observed in the along-track direction. This is probably due to the over-simplification of the Earth rotation model.

Conclusion:

The TOPEX/Poseidon experiment demonstrated the good operational behavior of the on-board orbit determination software and confirmed that the orbit accuracy requirements of SPOT 4 can be met.

This experiment also demonstrated the interest of near real-time ground based processing of the DORIS data to assess the performance of the instrument, and eventually, to conduct spacecraft operations.

Current tests on 1750 emulators confirm that results of similar quality should be obtained in real-time on-board of the SPOT 4 spacecraft. If the DORIS instrument on-board of TOPEX/Poseidon had been equipped with the orbit determination card, position information with an accuracy of about 50 m rms would have been available to the spacecraft and all of its instruments since the turn-on of DORIS.

Therefore, mission designers who plan to use DORIS can count on the availability of reliable position information on-board of the spacecraft.

Acknowledgements

The authors wish to thank D. Mainguillon of the CNES Ada group for his patience in helping them apply the concepts of Object Oriented Design to this project and for guiding them through STOOD. They also wish to thank E. Cazala-Hourcade and L. Maisonobe who cheerfully provided them with the support needed to use MERCATOR, both at CNES and at JPL.

References

1. NOUEL F. *et al*, DORIS: A Precise Satellite Positioning Doppler System, Astrodynamics 1987, in *Adv. Astron. Sci.* 65, Solder et al (eds), 311-320, 1988
2. BERTHIAS J.-P., JAYLES C., Real-time orbit determination on board of SPOT 4, proceedings of the IFAC workshop on spacecraft automation and on-board autonomous control, Darmstadt, Germany, 14-16 Sept. 1992
3. TAVERNIER G., CAMPAN G., MERCATOR: Methods and Realization for Control of the Attitude and the Orbit of spacecraft, SpaceOps 92, *this volume*
4. FRAUENHOLZ R., Operational Navigation Support for the TOPEX/Poseidon Mission, SpaceOps 92, *this volume*
5. NOUEL F. *et al*, Precise orbit determination with the DORIS/SPOT 2 system: first results, proceedings of the ESA symposium on Spacecraft Flight Dynamics, Darmstadt, Germany, 30 Sept.-4 Oct. 1991

TOPEX/POSEIDON PRECISION ORBIT DETERMINATION PRODUCTION AND EXPERT SYSTEM

N94-23920

Barbara Putney

NASA/Goddard Space Flight Center
Greenbelt, Maryland 20771 USA

Nikita Zelensky and Steven Klosko

Hughes STX
Lanham, Maryland 20706 USA

ABSTRACT

TOPEX/Poseidon (T/P) is a joint mission between NASA and the Centre National d'Etudes Spatiales (CNES), the French Space Agency. The TOPEX/Poseidon Precision Orbit Determination Production System (PODPS) was developed at Goddard Space Flight Center (NASA/GSFC) to produce the absolute orbital reference required to support the fundamental ocean science goals of this satellite altimeter mission within NASA. The orbital trajectory for T/P is required to have a RMS accuracy of 13 centimeters in its radial component. This requirement is based on the effective use of the satellite altimetry for the isolation of absolute long-wavelength ocean topography important for monitoring global changes in the ocean circulation system. This orbit modeling requirement is at an unprecedented accuracy level for this type of satellite. In order to routinely produce and evaluate these orbits, GSFC has developed a production and supporting expert system. The PODPS is a menu driven system allowing routine importation and processing of tracking data for orbit determination, and an evaluation of the quality of the orbit so produced through a progressive series of tests. Phase I of the expert system grades the orbit and display test results. Later phases undergoing implementation, will prescribe corrective actions when unsatisfactory results are seen. This paper describes the design and implementation of this orbit determination production system and the basis for its orbit accuracy assessment within the expert system.

1. INTRODUCTION

The TOPEX/Poseidon Mission requires unprecedented orbit modeling to achieve its ocean science goals. (Ref.1) The satellite orbit must be determined with an RMS radial accuracy of 13 centimeters. This is an extremely stringent accuracy requirement

for a satellite of this shape and altitude. T/P is in a circular orbit at 1340 Km altitude. It is inclined with respect to the equator by 66 degrees. It is a large satellite with a 28 m² single solar panel and weighs nearly 2500 kg. Three precise tracking systems are supporting T/P. Satellite laser tracking (SLR) is the baseline tracking network for the NASA portion of the mission. Currently, there are more than 30 worldwide laser stations providing data on T/P. DORIS is a one-way ground-to-satellite dual frequency Doppler tracking system developed in France which provides tracking data from a worldwide network of over 45 ground beacons. Unlike SLR, the DORIS systems are unaffected by weather and provide nearly continuous monitoring of the T/P orbit. Tracking of T/P by the Global Positioning System (GPS) through an inter-satellite signal is an experimental tracking mode being tested on T/P. The GSFC PODPS uses the complete set of laser and DORIS data to support its orbit computations. The altimeter data (which directly measures the height of the satellite above the ocean surface) is used within PODPS as an independent reference for radial orbit accuracy assessment.

To meet the orbit accuracy requirements for T/P, GSFC has produced improved gravitational models (Marsh et al., 1988 (Ref.6), 1990 (Ref.7); Lerch et al., 1992 (Ref.5)) and developed appropriate non-conservative force models taking into account the complex satellite form of the T/P spacecraft (Marshall et al., 1991 (Ref.8), 1992 (Ref.9)). There has been a great deal of progress made in both areas as verified by orbit analyses using the first two months of the T/P data now available. The routine determination of orbits having sufficient accuracy in a expedient fashion (25 working days after the completion of each 10 day repeat cycle of the satellite) necessitated the development of a production system. In order to

satisfy these accuracy and time of delivery requirements, we have developed a production system called the Precision Orbit Determination Production System (PODPS) which performs the production task consistently, under strict configuration controls, and which has an expert component to insure that the required accuracy is achieved. If for some reason an unsuitable trajectory is produced, these orbits will be reprocessed in an attempt to remedy the problem. If however, the orbit cannot be improved to yield a trajectory which passes the quality control tests, the orbit is flagged to warn the ocean science community of the suspected degraded accuracy obtained during this cycle. This paper will describe the approach and design of PODPS which has been implemented to achieve its goals.

2. THE PRODUCTION SYSTEM

The objectives of this system are to automate, consolidate, and strictly control those functions necessary for routine determination of precise orbits : data import, data processing, orbit generation and evaluation, and archive. The system is designed to be portable. It is built in modular form; this design enables new requirements to be implemented as we gain experience with the T/P mission data sets. As they become defined, new capabilities are also easily implemented; therefore, the system has a wide variety of uses beyond the specific needs of the T/P mission.

The orbit determination package which is used is the GEODYN II program (Eddy et al., 1989 (Ref.3)) which is a state-of-the-art system developed at GSFC. GEODYN I and II have been used to perform precision orbit determination for over 20 years at GSFC. The software has been evolving over time to include sophisticated models for the forces arising from the static gravitational field, earth and ocean tides, relativity, atmospheric drag, direct solar radiation, Earth albedo, third body gravitational and satellite thermal imbalance effects which perturb the satellite's motion. The program uses sophisticated models to address the motion of the Earth within an inertial frame, and the motion of an observer on the Earth's surface (station tectonic, earth tidal and ocean tidal loading motions). Since 1982, a major undertaking has been the improvement of existing models of the Earth's gravitational field to meet the factor of five improvement required for T/P. The GEODYN II program has been the workhorse for this large effort. A gravitational field, represented as a spherical harmonic expansion complete to degree and order (70,70), has been determined from 31 satellites

using over 2.5 million observations. This T/P pre-launch model is called the JGM-1 (Joint Gravity Model), being a model jointly determined by Goddard Space Flight Center and the Center for Space Research at the University of Texas in Austin. The error in the TOPEX/Poseidon orbit due to the static gravity field cannot exceed 10 centimeters RMS in the radial direction in order to meet the 13 centimeter RMS radial orbit accuracy required by the mission. JGM-1 will be updated through the incorporation of T/P SLR and DORIS tracking data producing JGM-2 (under development) to provide the gravity modeling for T/P throughout its observational phase of the mission. Consequently, following gravity and non-conservative force model verification, we can use the GEODYN II program with confidence for the production of the POE files.

2.1 Hardware Configuration

In anticipation of the rapid advances in hardware technologies, the PODPS system was designed to be portable across a wide range of computer environments. An early decision was made to create a menu-based system that would be written in Fortran so that it would port easily to different hardware platforms. This turned out to be an important consideration since our computational environment has evolved considerably over time, and our current configuration will likely continue to change in the near future. When we started our preparations for the T/P Mission, we were using a Cyber 205 super-computer supported by a "front-end" Amdahl V-7 running the IBM MVS operating system. Another Amdahl, the V-6, was concurrently running the VM operating system while an IBM 3081 ran both the MVS and VM operating systems. The GEODYN II program is really three programs, (1) a flexible tracking data reformatting package (TDF), (2) the GEODYN IIS package which generates data and interface files describing the desired orbit solution and (3) the GEODYN IIE system which is the computational engine performing the orbit determination solution. From 1982 onward, GEODYN IIE ran primarily on the Cyber, and through a redesign of code, was made to benefit from an extensive vectorization of its algorithms. The remainder of the GEODYN system ran on the AMDAHL or IBM using the MVS system. In 1986, the Cyber 205 and the Amdahls were replaced with a CRAY YMP super-computer.

The CRAY YMP is now our orbit computation hardware platform. Since PODPS is an operational

system, a suitable backup capability (although with less "horsepower") is provided by the IBM 9021 which is the current front-end to the CRAY. At this point we are beginning to hear that the MVS system on the 9021 will likely be replaced during the mission timeframe with a UNIX operating system. This would bring the front-end into closer compatibility with the UNIX operating system on the CRAY which is running UNICOS. We have decided to integrate workstations into the hardware configuration of the PODPS system now that they have demonstrated enough CPU power to meet our needs. This will also provide hardware stability for the remainder of the T/P timeframe. Two HP 730 are now being phased into the system to provide backup capability and provide additional file service to the PODPS. Therefore, while undergoing development, the PODPS system had to be ported several times across systems as the computer environment around us changed. Currently, our prime system is on the CRAY and our backup system is on an IBM 9021 computer. We are in the process of porting the system to the HP 730s. Our combination of Fortran and UNIX will allow an easier transition, but not one completely free of pain. Planning for portability has been essential. Presently, our prime system is running on the CRAY YMP using the IBM 9021 as the file server. The next transition will be to make the HP both the file server and the platform for running the TDF and GEODYN IIS systems. At present, we foresee retaining the CRAY YMP for the computationally intensive GEODYN IIE work.

2.2 PODPS Software and Data Management Description

The menu program is written in Fortran 77. The system currently contains about 50 individual programs and requires over 100 files for each arc. The menu program is the driver for the system. The maximum depth for the menu is four levels. The main menu has six functions: Arc Selection, Schedule, Data Acquisition and Reformatting, Data Reduction and Orbit Verification, POE Generation and Archive and File Management Utilities. The Arc Selection allows a technician to choose the satellite and arc that they want to work with. The Schedule function displays the 22 work day schedule of tasks required in the processing of the specific POE arc. The Data Acquisition and Reformatting function imports laser, DORIS Doppler, altimeter, solar and magnetic flux, polar motion/earth rotation and SLR station eccentricity values. The user is transferred automatically to the NASA/Crustal Dynamics Data

Information System (CDDIS) Vax computer where all the data sets are gathered. The SLR, DORIS, and eccentricity data are already on the CDDIS, the fluxes are obtained from a NOAA Vax in Colorado, and the polar motion and earth rotation values (which are being determined from a concurrent analysis of LAGEOS SLR data) are obtained from a CRAY YMP at the University of Texas in Austin. The set of station positions for this cycle consistent with the Earth orientation series are assembled. The data are reformatted. Normal points are created for laser data for those stations that do not supply station generated normal points. After the staging of this information, the real analysis can begin.

The Data Reduction and Orbit Verification function is the main workhorse of the system. Several types of orbit determinations are performed. First one starts with the POE data reduction. Each orbital solution is followed by the execution of a program called REP (Residual Analysis Program) which analyzes the orbital residuals and determines estimates of measurement and timing biases which reflect both orbit and data problems. REP creates delete cards for spurious observations to remove them from future POE data reduction runs. There is also an interactive graphics program called IRE (Interactive Residual analysis) where one can look at the residuals graphically and create delete cards for points not desired in future iterations. These files are included automatically in subsequent orbital solutions bringing the data verification aspects of the process into a state of convergence. When the orbit seems to be satisfactorily converged, the Overlap and High Elevation data tests are performed. The Overlap test uses 5 day overlap orbit determinations with the previous and next arc and compares the overlap orbits with the converged orbit. This checks for consistency of the orbit from arc to arc. The High Elevation test deletes all SLR passes that have elevations above some value, nominally 60 degrees, and recomputes a new orbit. The orbits are then compared and the omitted data residuals from the high elevation passes are used to project radial orbit error at the times of these independent data. The final tests use the altimetry data from the TOPEX/Poseidon satellite. The altimetry data residuals are computed from the converged POE orbit determined using SLR and DORIS and evaluated both geographically and in their temporal characteristics. Altimeter crossovers are determined for the arc and residuals are computed for them. At each step the desired information is being collected in worksheets for application of the expert system criteria that are to be applied to the arc. This

includes displaying the worksheets created for each of these test activities.

The POE Generation function creates the Quick Look POE and the final POE. The Quick Look POE can be created in less than 1 week after the arc has been started. At this point, the data import has not been completed nor the orbit verified. The final POE is sent to JPL for inclusion on the altimeter Geophysical Data Records (GDRs), CNES for comparison with their orbits, the University of Texas for verification, and the University of Colorado for non-conservative force modelling evaluation. The Archive function archives the arc enabling its reanalysis should it become desirable.

The File Management Utilities help to manage the information of the system. The file structure and gathering of the appropriate information from the proper module is monitored automatically by this subsystem. Unfortunately, this is the most difficult element to make portable. Each computer has its own idiosyncrasies for disk and file management.

The above is a quick synopsis of the system which has been designed for easy use and provide the information for evaluation by the expert system. The system, force modeling and satellite sensor complement is currently undergoing extensive evaluation and verification. A workshop is scheduled to determine the appropriateness of having T/P enter its final Observational Phase. The production part of the mission begins after the Science Verification Workshop which is scheduled for February 1993 where this assessment will be made.

2.3 The Load Test

The PODPS system is being built in stages. Each stage includes a period of testing. One test that was performed before launch was the Load Test. This was a test using the SLR data taken on the Ajisai satellite. It checked the laser data sufficiency and availability timeline. It checked our ability to determine an orbit and produce a POE. The POE was distributed for format and interpolation checks. One month of Ajisai data was processed. The PODPS system used was completely housed on the CRAY and was used to evaluate the system in place at the time of T/P's launch. Unfortunately, Ajisai does not have altimeter or DORIS data so these parts of the system could not be checked using coincident actual observations. We nevertheless obtained orbits meeting or exceeding the accuracy goals we set for

Ajisai during these tests. This testing also was useful in locating weaknesses in our file system and problems when the file server was not acting properly. We modified the system based on this experience.

2.4 The Expert System

The objectives of the expert system are to collect and present information required for the evaluation of the POE orbit quality. The aim is to automate this processing. It is also desired that the system evolve based on lessons learned.

Having the production system output a condensed summary of results called worksheets at each major operation is a big step towards creating an expert system. The actual process of creating the system is to identify what information is needed, where it can be obtained and how it should be evaluated. In essence, this is an effort to formalize and automate the experimental knowledge of our orbit analysts into an automated computer system.

Over 200 quantifiable criteria related to orbit performance have been identified in all of the functions described above. These criteria were chosen because it is anticipated that the accuracy of the orbits is sensitive to them, it is measurable and its measure will predict the accuracy of the orbits by itself or in conjunction with other criteria. Our most experienced orbit analysts created the lists based on their experience in assessing orbit accuracy. It is anticipated that some of the criteria will be eliminated and others added as a result of on-orbit experience with T/P. A "straw-man" value for each criteria was determined by performing error analyses. In addition, the criteria were evaluated during the Load Test. The modules of the production system were modified to output the information to worksheets. Once the required information is collected, software was written to perform the analysis and score each arc. The passing or failing of a specific test criteria is classified as either a deficiency check or a test of critical failure. Suggestions are given to the technicians as to actions to be taken in cases of failure and possible causes for the anomalous result.

The current expert prototype system applies the most important 120 criteria to evaluate the quality of the candidate orbit. A pass/fail threshold value determined from pre-launch simulations is assigned to each of these criteria. Since the criteria vary in significance, a pre-launch weight has also been assigned so that an

overall score for the orbit accuracy can be determined. The choice of criteria, criteria pass/fail values, and criteria weight values are expected to evolve as we gain experience processing TOPEX data. Nevertheless, this extensive automated testing process ensures detailed scrutiny of each of the resulting orbits.

The criteria are grouped into seven test categories: Data Import, Data Reduction, tracking data Residual Analysis, High Elevation Pass, Orbit Overlap direct comparison, Altimeter Range residual analysis, and Altimeter Crossover residual analysis. A final sanity check is also made on the POE just before export. Figure 1 provides a qualitative overview of the information content of the various orbit quality tests. As shown, the use of independent altimeter data is expected to contain the most information concerning orbit quality. The challenge is to extract the orbit quality information from altimeter data. No one test can provide sufficient conditions to fully assess a given orbit's quality; however, certain criteria, if failed, cause the orbit to be deemed of insufficient accuracy for release to the project. Orbit accuracy will be estimated using the combined results from all the tests, and will be based on our understanding of the nature of geographically correlated and time-varying orbit error.

Figure 1

TOPEX PODPS INDICES OF ORBIT QUALITY ASSURANCE

TESTS	INTERNAL CONSISTENCY	ORBIT PRECISION (TIME VARYING)	ORBIT PRECISION (GEOGRAPHICALLY CORRELATED)	ORBIT ACCURACY (VS PRECISION)
Data reduction statistics	**			
Laser residual analysis	***	*	*	
Orbit overlap comparison	****			
Altimeter crossover analysis		***		*
Altimeter residual analysis		****	***	**
High elevation pass data subset			**	***

* Weak
*** Strong Extremely limited distribution

An overview of the nature of the testing follows. The Data Import criteria provide a sufficiency and sanity check on the tracking, polar motion, solar flux, and other data collected for input to the data reduction step. For example, if there is a sufficiently large gap in the tracking data, the operator will be alerted to this condition and advised to modify the parameterization of the non-conservative force model parameters in the data reduction. The data reduction statistics output from GEODYN, such as the rms fit on the data, are compiled and displayed. Pre-launch simulations have shown that the rms of fit on the data will be about 1/2 the rms of the total orbit error. In a separate program, the timing and offset bias is

estimated for each pass of the tracking data residuals. The Residual Analysis statistics aid in data editing and provide the first estimate of the along-track orbit error. The High Elevation Pass test takes advantage of the high precision offered by laser tracking. SLR high elevation passes, removed from a subset solution, provide an independent absolute estimate of the radial orbit error. However precise, this orbit test is limited both temporally and spatially over the small subset of the high elevation passes selected. The Orbit Overlap test directly compares the trajectories from two 10-day solutions over 5 days of overlapping data. This is an excellent test of orbit precision, but since common errors will cancel, cannot directly estimate orbit accuracy but checks consistency with a high degree of confidence.

Altimeter data offers a direct measurement of the satellite height relative to the sea surface. In order to use altimeter range for evaluating the radial orbit accuracy, the distance from the sub-satellite sea surface to the geocenter must be modeled. Although altimeter precision is good to 2 cm, considerable error is introduced modelling several oceanic features including the mean sea surface height (geoid) and sea surface variable topography due to tides and changes in the boundaries of the current systems. The sum of modeling errors is expected to total about 20 cm rms over a 10 day period. One may ask how can this measurement be used to test orbit accuracy to 13 cm ? There are two means. Most of the 20 cm is due to the geoid error which varies only spatially, and, as the satellite samples it, at a higher frequency than orbit error. Altimeter crossover measurements are formed by differencing two altimeter ranges taken at the intersection of an ascending and descending pass. The crossover residual thus contains only time-varying error, of which orbit error is by far the major component. In the second approach orbit error can be estimated from altimeter range residuals using an empirical function with the following form:

$$\Delta h = a + (b \cos(\omega_t) + c \sin(\omega_t))$$

(bias) (1 cycle/rev)

$$+ d \cos(2\omega_t) + e \sin(2\omega_t)$$

(2 cycle/rev)

$$+ f (t-t_{mid}) \cos(\omega_t) + g (t-t_{mid}) \sin(\omega_t)$$

(bow tie)

Where:

ω_t = orbit frequency

t = time from the arc start

t_{mid} = time of the arc midpoint from arc start

This function contains the dominant error signal expected for radial orbit error, containing the familiar bowtie, once-per-rev, and twice-per-rev terms (see Colombo, 1984 (Ref.2); Engelis, 1987 (Ref.4)).

For an example, Figure 2 shows the Altimeter Range Expert System Summary using 8 days of actual T/P data acquired during its Cycle 1 altimeter data to evaluate a preliminary POE. The shape of the estimated orbit error is shown in Figure 3. From these and other tests the orbit error is estimated to be at or slightly higher than the 13.2 cm allowed. These results are very encouraging since they are based strictly on pre-launch models and considerable improvement is expected following gravity and non-conservative force model tuning using the T/P tracking data.

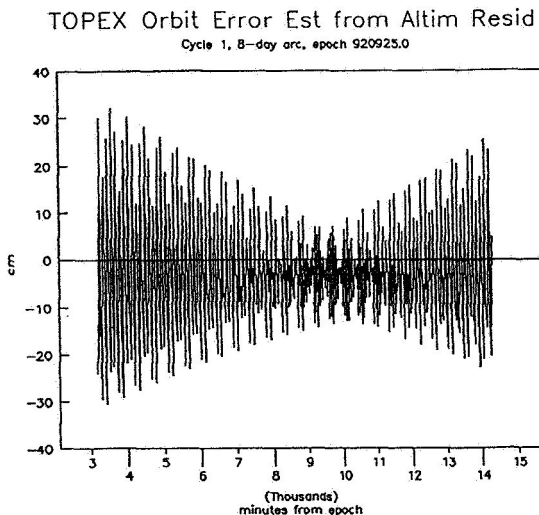
Figure 2

SUMMARY SUBSECTION EXPERT SYSTEM ALTIMETER RESIDUALS RESULTS

MODULE TEST#	TEST DESCRIPTION	UNITS	RESULT	CRITERIA P/F	SCORE
RESIDA 1	RMS OF ERROR FUNC.	CM	12.467 LE 10.000 F		0.2467
RESIDA 2	AMPLITUDE OF 1 C/R	CM	10.991 LE 7.000 F		0.5701
RESIDA 3	AMPLITUDE OF 2 C/R	CM	9.850 LE 3.000 F		2.2834
RESIDA 4	AMPLITUDE ARC MODU C/R	CM	10.508 LE 7.000 F		0.5012
RESIDA 5	RMS RESID (ALT)	CM	25.366 LE 25.000 F		0.0146
RESIDA 6	PERCENT ALT EDITS	%	1.325 LE 10.000 P		-0.8675
RESIDA 7	ALTIM TIME ERROR	MS	-4.819 LE 5.000 F		-0.1811
RESIDA 8	ALTIM BIAS	CM	-3.237 LE 15.000 P		-11.7631
RESIDA 9	RMS ALTIM: NE	CM	29.414 LE 25.000 F		0.1766
RESIDA 10	RMS ALTIM: NW	CM	20.563 LE 25.000 P		-0.1775
RESIDA 11	RMS ALTIM: SE	CM	23.428 LE 25.000 P		-0.0629
RESIDA 12	RMS ALTIM: SW	CM	19.536 LE 25.000 P		-0.2168
RESIDA 13	RMS RESID POST-PT	CM	21.853 LE 15.000 F		0.4568

COMPRSED OF 13 TESTS
SHOWING 6 PASSED AND 6 FAILED

Figure 3



3.0 CONCLUSION

A production and expert system has been created to produce highly accurate orbits for the TOPEX/Poseidon Mission. The radial accuracy requirement is 13 centimeters RMS. A prototype of the system is now operating. It is written in Fortran

77 using UNIX scripts and designed to be portable. This system offers us advantages for it both produces and quality checks the orbits for T/P. It can locate poorer performance for a particular arc, allowing special attention to be paid to its improvement. It also allows us to alert the science community when an orbit's accuracy is suspect. The uniform approach is anticipated to provide uniformly accurate orbits. This system has wide ranging applications for other missions requiring precision orbit determination.

ACKNOWLEDGMENTS

The design and development of the PODPS production and expert systems is the talented work of Ken Rachlin, Craig Johnson, and Terry Williams of Hughes/STX. Another very important orbital expert is Ron Williamson of Hughes/STX. Many others of the TOPEX/Posiedon PODPS team contributed to the system. The work is funded by the TOPEX/Posiedon Project of NASA.

4.0 REFERENCES

1. Born, G.H., et al., "Accurate measurement of Mean Sea Level Changes by Altimetric Satellites", J. Geophys. Res., 91, 11,775-11,782, 1986.
2. Colombo, O.L., "Altimetry, Orbits and Tides", NASA Tech. Memo. TM-86180, 173pp., 1984.
3. Eddy, W.F., et al., "GEODYN II System Description: Vol 1-5", prepared for Space Geodesy Branch, GSFC, September, 1989.
4. Engelis, T., "Spherical Harmonic Expansion of Levitus Sea Surface Topography", Rep 385, Dep. Geod. Sci. and Surv., Ohio State Univ., Columbus, 1987.
5. Lerch, F.J., et al., "Geopotential Models of the Earth from Satellite Tracking, Altimeter and Surface Gravity Observations", GEM-T3 and GEM-T3S", NASA Tech. Mem. 104555, January, 1992.
6. Marsh, J.G., et al., "A New Gravitational Model for the Earth from Satellite Tracking Data: GEM-T1," J. Geophys. Res., 93, B6, pp. 6169-6215, 1988.
7. Marsh, J.G., et al., "The GEM-T2 Gravitational Model," J. Geophys. Res., 95, B13, 22,043-22,071, 1990a.
8. Marshall, J.A. et al., "Modeling Radiation Forces Acting on Satellites for Precision Orbit Determination", AAS-91-357, AAS/AIAA Astrodynamics Conference, August, 1991
9. Marshall, J.A., et al., "Modeling Radiation Forces Acting on TOPEX/Poseidon for Precision Orbit Determination", NASA T.M. 104564, June, 1992.

A Real-Time Navigation Monitoring Expert System for the Space Shuttle Mission Control Center

N94-23921

Lui Wang, Malise Fletcher
NASA Johnson Space Center
Houston, TX 77058

ABSTRACT

The ONAV (Onboard Navigation) Expert System has been developed as a real time console assistant for use by ONAV flight controllers in the Mission Control Center at the Johnson Space Center. This expert knowledge-based system is used to monitor the Space Shuttle onboard navigation system, detect faults, and advise flight operations personnel. This application is the first knowledge-based system to use both telemetry and trajectory data from the Mission Operations Computer (MOC). To arrive at this stage, from a prototype to real world application, the ONAV project has had to deal with not only AI issues but operating environment issues. The AI issues included the maturity of AI languages and the debugging tools, verification, and availability, stability and size of the expert pool. The environmental issues included real time data acquisition, hardware suitability, and how to achieve acceptance by users and management.

Key Words: Expert Systems, CLIPS, Knowledge Based System Verification & Validation, Onboard Navigation, Mission Control Center

1. INTRODUCTION

The Onboard Navigator (ONAV) is a support room position that assists the Guidance and Procedures Officer (GPO) in the Mission Control Center. The ONAV's principal responsibilities are to monitor the health of the onboard state vector and navigational aids that are used to update the state vector and make recommendations to the GPO for actions to maintain the state vector. The state vector represents the orbiter's position and velocity at

a given time and is used in various guidance and control functions on the orbiter. The state vector is propagated using the inertial measuring units (IMU's) and is updated by a variety of navigation sensors through a kalman filtering scheme. The sensors used are dependent on the phase of flight. ONAV supports three phases of flight; ascent, entry and rendezvous. During ascent only the IMU's are used; during entry the IMU's along with the tactical air command and navigation system (TACAN), air data transducer assembly (ADTA), a drag altitude model and the microwave scanning beam landing system (MSBLS) are used. Rendezvous uses the IMU's, star trackers and the rendezvous radar. Basically, the ONAV job entails monitoring several digital displays that provide information on the status of the various navigation sensors and the onboard state vector. Approximately 180 parameters are normally monitored. The ONAV, based on this information, makes recommendations ranging from incorporating the sensor data into the onboard filter to recommending a ground derived state vector be uplinked to the onboard system. To be able to perform this function the ONAV must have an understanding of how the onboard software operates, how the sensors function, crew procedures, and a knowledge of kalman filtering. In addition the ONAV must be able to rapidly determine what information on the displays is important, which varies as a function of time, what failures have occurred, and what sensors are available.

2. OBJECTIVES AND GOALS OF SYSTEM

The objective of the system is two fold; one is to use the system as a console assistant and the other is as a training tool. Currently each phase of the mission requires two ONAV's for support; a lead and an ONAV2. The lead's

responsibilities are to coordinate all information from the console, communicate with the GPO and make all pertinent decisions. ONAV2 logs information and provides backup to the lead. As the system matures and both the GPO's and ONAV's become more confident with it, the goal is to eliminate the ONAV2 position. Also the system should enhance the quality of mission support. This is being accomplished in a variety of ways. The first method is the automatic logging capability of events and recommendations with both an altitude and a time tag that the system generates. Previously the ONAV's would log these events by hand and if several critical problems occurred at once it became difficult to keep up. The system will eliminate this problem. Another enhancement is the incorporation of color graphics in the user interface. This provides an easier to read screen and significantly aids in problem recognition and resolution by providing trend analysis.

2.1 Console Assistant

Probably the most critical enhancement provided by the system lies in its ability to monitor all of the sensors at all times. At certain periods, one sensor may be more critical than others. As a result, the ONAV's attention tends to focus on the status and the functions of that one sensor and loses track of the others. The expert system, however, has the ability to continuously monitor all of the available sensors while still providing increased focus on the specific sensor that is providing critical data. Over 600 parameters are monitored by the entry system, 400 parameters for the ascent and 425 parameters for the rendezvous system. This function has already proven to be highly valuable during flight simulations conducted by Mission Operations Directorate (MOD), when the system alerted controllers to a secondary problem.

2.2 ONAV Trainer

The second objective for the system is to use it as a training tool. Currently, training an ONAV2 takes over a year. An additional nine months is then required for ONAV1 certification. A major impact to the length of time that the training process takes is the number of simulation hours available. A new trainee can

expect, at most, four hours a week in simulation time. The expert system will lessen the trainee's dependence on simulations by providing the capability of running training simulations in an office environment. The biggest advantage of the use of the expert system as a training tool, however, lies in the establishment of the rulebase and its knowledge capture feature. When running as a training tool, the expert system will be equipped with a knowledge browser and desired decision rationale. This will allow the trainee to assimilate the information at his own pace and as a result he will not be as dependent on having an experienced person available to answer questions. Finally, by establishing a single set of agreed to rules the certified rulebase will provide a source of consistent training.

3. IMPLEMENTATION

The development of the rules for the ONAV expert system was a joint effort between the domain experts of the MOD and knowledge engineers from the Mission Support Directorate (MSD). The ONAV domain knowledge was captured through a series of meetings where the ONAV experts presented information to the knowledge engineers in a classic classroom setting. Initial meetings focused on the system functional overview. The results of the meetings were converted to different design strategies and prototypes, which could be critiqued by the experts. Prototype development then, served as the feedback mechanism for the meetings. During the development phase, when the knowledge engineer understood the basic operations, they would generally develop the agenda for the meetings on a particular topic and the experts provided the detail knowledge. The meetings were scheduled two to three hours once a week for the first three months, and after were held less frequently. By the end of six months the first prototype was completed.

3.1 Functional Design

It turned out that the ONAV normal task functions were fairly straight forward; however, detecting anomalies in the subsystems and recommending proper response required detailed expert knowledge. The overall system

design was completed in the first year of the development phase. This structure resulted from the basic nature of the ONAV task and from the modularity guidelines of any good system engineering approach. Four functional components of the expert system can be identified: (1) Fact assertion, (2) monitoring, (3) analysis, and (4) output. In addition, there is a fifth non-expert system component that is a part of the overall ONAV system called "data preparation".

3.2 Data Prep Design

The data preparation (data prep) component receives information from the operational environment and performs the following:

- Collects the information required by the expert system
- Performs any computations required on the data
- Filters and transforms that data into a form suitable for the expert system.

Often this component has been over looked, since this module contains no "AI" component. In fact, this component serves as an important safety net to ensure real-time (600 parameters every two seconds) system performance. An real-time monitoring system is worthless, if it can not keep up with the data streams. In addition to the real-time constraint, there is no reason for the expert system to examine every data point; especially, when the expert system is mostly interested in accessing the qualitative status of the LRUs and the overall performance of the entire navigation system (i.e. the onboard navigation Kalman filter). Thus, it is the data prep's responsibility to deduce the higher order abstractions or concepts (i.e. large imu biases, growing tacan residual, etc..) from a stream of instantaneous numerical data values. It turns out that the higher level abstraction can be modeled from a set of well defined mathematical and statistical functions which are more appropriately implemented using conventional programming techniques. In short, the data prep component converts the numeric instantaneous data points to a set of symbols which reflect the state of the environment. The fact assertion component simply takes the prepared data and puts that data required by the expert system into the fact base. The monitoring component generates intermediate

conclusions and statuses of the individual subsystems that ONAV observed and manages. This component simply detects environment state changes and pass them on to the next component. The analysis component performs an overall assessment of the current status, (i.e., makes use of the data from the monitoring phase) confirms the subsystems status, and/or recommend actions to prevent further degradation of the state vector. The analysis phase is the heart of the knowledge base, because this component contains the captured the ONAV expertise. Specific knowledge implementation such as letting TACAN take out the state error rather than doing an uplink to the orbiter, or deselecting the line replaceable unit (LRU) when the onboard data bus has a commfault, were programmatically captured, and kept. The output component controls the sending of the notices and/or recommendations to the ONAV expert system console.

3.3 Rulebase Design

The ONAV rulebase is grouped based on four processing phases which is described in the previous section. The rule execution is governed by a set of phase control facts and phase control rules. Using this control structure, one can easily direct the program flow of the different processing modules. This approach is used throughout the ONAV knowledge base. For instance, the different navigational sensors, such as the IMU's, TACAN, ADTA, and MSLS for the Entry Nav. and the IMU's, and Rendezvous Radars and Star Trackers Radars for the Rendezvous Nav. of the Shuttle operation, are separated by different control facts. Thus the sensor modules are nicely grouped and the inter module rule interactions are kept and continuously operate in an opportunist data-driven fashion. In addition, the modular rule set enable the development team to support an incremental development process. Since, rules are changed quite frequently to accommodate the different control strategies for modeling the ONAV flight controllers. The modeling of the flight controllers' behaviors is quite unique by its own right. Only through the power of rulebase programming, one can easily encapsulate the set of heuristic in the form of rules.

3.4 Interface Design

As for the look and feel of the system, the philosophy was to keep it as simple and clean as possible. To achieve this a pop and shoot method was chosen. This method consisted of a series of pop-up menus or toggle switches activated by the mouse. This provides a format that is easy to read and allows the operators to keep their eyes on the screen, a required feature in the highly dynamic environment of orbiter high speed operations. Messages appear in windows that can contain up to 20 messages and are scrollable. Status and quality lights take full advantage of color and also contain text to insure that the operator understands the meaning. In addition, a real-time plot package is developed to augment and support the expert system program. For instance the expert system issue an off nominal event, due to a drifting IMU. When this situation is happening, the console operator can quickly pop up the appropriate plots to analysis the data.

3.5 Data Retrieval Design

As a real time console assistant, the expert system requires real time telemetry and trajectory data from the Mission Operations Computer (MOC), using a data stream called Generalized Data Retrieval (GDR). The GDR data is actually retrieved by a program called GDR_driver, which reads the data from the MOC and writes it into a shared memory segment. This shared memory interface is based on a model developed at JSC for use by real time applications (ref. Workstation Application Interface to Data Source Interface Agreement). This model is generic and modular and it provides a standard mechanism to access different data sources. By using this type of architecture, the expert system is able to acquire data not only from the real-time data source but also acquire data from a repository archive library. The library of test cases do not only sever as a way for doing software verification but also can be used to train flight controllers in their office environment.

4. CERTIFICATION

The certification/verification scheme devised for the system was established at three different levels. The first level consists of the component level; data acquisition (GDR), data prep, rulebase, plots and the interface. Its intent is to insure that the program is functionally correct, i.e.. that data is correctly gathered, that the rules fire if presented with appropriate actions and events, and that results are correctly displayed to the screen. The second level consists of integrated system testing and overall certification. The intent of this level of testing is to insure that the various constituent parts of the system function in an integrated manner and, more importantly, that the recommendations made by the system accurately model the expert's mental concepts of the state of the system and are the correct responses to environmental changes. The third level consists of an integrated workstation certification. This last level is accomplished because of the critical nature of the workstations and is to insure that this application will not interfere with any other workstation or mainframe applications. Finally, the entire process is documented and presented to the MOD Certification Board.

4.1 GDR & Data Prep Certification

For the GDR certification, data is collected during a sim at several time points. Hardcopies are requested from the operations computer (MOC). For each time point, data is compared with the MOC to the data received at the workstation. Once the GDR data has been verified, data prep module is verified and validated in a similar manner. The next step in component testing was the verification and validation of the rulebase, which has been the most difficult of the tasks.

4.2 Rulebase Certification

To attack the problem of verification, each rules is traced and checked manually with the experts' requirements which were decided in the flight technique meetings. However, the main problem is to devise a method to validate these flight procedures along with the rest of the system. The existing literature has little to

offer on verifying and validating large rulebases. The final decision, after much debate, was the construction of an error matrix that contains 48 errors that a certified ONAV would have to be able to deal with to be certified. For the rulebase to be declared verified, the system must act correctly for each of the errors under several conditions.

The method that was chosen to verify the rulebase has largely been dictated by the source of test data. The optimum method would have been to run a nominal case and then introduce a specific error in each of the following runs. Due to the complexities of the problem environment, the only source of data for the system is simulations or missions. However, the goal of integrated simulations are not gearing specifically to train ONAV flight controllers, therefore the training area as to what errors occur in a sim can not be controlled easily. Consequently the verification is somewhat of a hit or miss operation. Hence, the data is logged and evaluated when they are available, not necessarily in a controlled, selected fashion.

When a sim run is selected, it is first delogged. In this manner actual copies of the controllers screens are obtained from the run. Next, an expert will go through the delog and determine what occurred in the case and what decisions and events the ONAV flight controller would have made and noted. The next step is for the expert to run the testcase through the expert system. At this time, any errors or discrepancies are noted. The expert then writes a log with the errors and discrepancies and what the correct action should have been. This is then passed to the software engineers, who in conjunction with the experts, work on correcting the problems. After the software engineers are done, then the corrected rulebase is passed back to the experts. The testcase is then replayed, and if all errors have been corrected and no new ones appear then the new rulebase is used in replaying all previous testcases. If everything checks out then the that testcase is accepted by all parties.

In the event any questions arise that concern procedures or requirements, the action is referred to the ONAV working group. The ONAV working group was formed, with representatives from all the functional ONAV

and software engineering areas, to focus the collective knowledge of the ONAV community and provide an authoritative method of resolving requirement conflicts or procedural debates. In this fashion controllers were provided input into the generation of the rulebase and a feeling that they know and understand what corrections have been made and the methodology that is incorporated in the rulebase. In addition the working group minutes serve to document the decisions made and provide the knowledge engineers a reference that they can refer to when creating/modifying the rulebase.

Once all the various components have been verified, the system will be ready for full certification. At this point the system will be treated in the same manner as any flight controller for certification. To obtain final certification, the system is run through a full set of entry sims with an GPO and ONAV1 monitoring its performance. The GPO and ONAV1 will then evaluate and determine final certification. Also during this phase the system will be checked to verify that it will not interfere with any other workstation application.

Since not only expert systems but also workstations represent a new way of console operations, the system will be implemented in three phases. The first phase, which was used during rulebase verification, is for a third controller to monitor the system behind the current operator. This allows us to run the system during actual sims and flights without conflicting with console operations. Now that the rulebase is verified, the system has been moved on console and is monitored by the ONAV2. This phase will allow us to do the final interface work and answer questions such as, is the system easy to read and does it convey all the information necessary for console operations. Once all operators become comfortable with the system then at that point it will be ready to replace the ONAV2.

5. STS-48 FLIGHT RESULTS

STS-48 was the first flight using the ascent and entry systems as a console assistant during real time operations. Overall the systems performed very well, correctly identifying all

the pertinent events during ascent and entry. During ascent the system correctly identified all the major mode transitions and the events; events including such items as SRB SEP, roll reversals and MECO. Pre-launch; eight seconds before lift-off, navigation gets initialized with a state vector, the expert system correctly called this event. During flight, as previously stated, the ONAV's main function is to monitor the shuttle's state vector performance. The expert system during STS-48 matched the ONAV's performance in monitoring the health of the vector. Also during this flight, one of the IMU's; IMU3, experienced a scale factor problem. The expert system not only identified a problem with the IMU but also identified the problem type.

5.1 STS-48 Detail Description

The expert system during entry also experienced good performance, even though this flight had some events occur that had not been seen during previous flights. Once again the system correctly identified all events and major mode transitions. Events during entry included sensor processing and TACAN lock on. During the STS-48 entry the runway was redesignated from KSC15 to EDW22, the expert system made the correct call in telling the GPO to change the ground radars to Edwards. The off-nominal event that occurred during this flight was that TACAN went into dilemma and self-test. Dilemma occurs when 2 TACAN LRU's are locked on but disagree with one another over a prescribed limit. The expert system correctly called the dilemma and self-test and also correctly identified which TACAN was the noisy unit. Proceeding through the entry profile; the next series of recommendations by the expert system consisted of calling for BFS transfers. The shuttle has dual software systems running concurrently, the primary and the backup. The backup state vector is not updated using the navigation sensors instead if the vector goes out of limits, a vector from the primary system is transferred over. The expert system correctly identified the BFS as out of limits and made the appropriate recommendation for a transfer and identified when the transfer was completed.

One of the main functions of the ONAV; as stated earlier, is to determine the health of

the nav sensors and recommend whether to use that information to update the nav state. After the TACAN dilemma had cleared and the noise settled down on TACAN; the expert system was able to reach the same conclusion as the operator. That conclusion was that the TACAN's were operating nominally and were safe to use. The same correct recommendation was also made for the Air Data system.

The bottom line from this first flight of the expert system was that the rulebase contains the essential knowledge of the ONAV flight controller. The system overall did an excellent job of conveying this information but several slight changes were suggested in the interface to allow even easier insight into the expert system information. These changes included making some messages less verbose and lengthening the time between some messages so that the screen does not fill up as quickly. The fact remains however that the ONAV expert system is, and will become even more so, a very valuable tool for controllers on console.

6. LESSONS LEARNED

From the project development perspective, the number one lesson is: Even though the expert system contains all the "flight rules" and corporate flight controllers knowledge, in order for this new technology (expert system) to become available for online flight operation, much time and efforts have to be devoted for system integration and acceptance from the user community.

Another interesting observation is that, time is one of the most critical elements on the project. Time provides the luxury of careful scrutiny of the system but if time is not carefully allocated it can rob the project of its momentum. The following problems represent areas that forced the project off its path and consequently slowed the development of the project.

6.1 Expert Availability Issues

As the name expert system implies, an expert is an integral part of developing an expert system. When this project first began, there were two ascent/entry experts and no certified rendezvous experts. For the projected flight rate at the time, six ascent/entry controllers

were needed and two rendezvous controllers. Due to the low numbers, the priorities of the available experts were flights, training, analysis and finally expert system development. So not only were there a limited number of experts, their availability was extremely limited also. At one point the only expert working on the entry system quit, leaving roughly a year and a half where no expert was available. These factors stretched the development out over a considerable length of time. This led to problems in the software engineering side, in that the programmers became restless with the lack of progress on the system. It consequently contributed in the decision by several programmers to leave the project. The expert system, in fact, had fallen prey to the very thing the system was trying to protect against, that of a lack of a knowledge base. Due to this problem, the luxury of time for a quick development of a prototype, at the start, was denied the project, consequently robbing the project of its needed momentum.

Due to the drawn out development time and turnover problems several different groups of experts worked on the entry system over time. This meant that the rulebase represented an inconsistent set of methodology and at times, due to limited documentation, a lack of understanding of why things were done the way they were. It was in response to this, that the ONAV working group was formed. Its function is to try to reach a consensus among current operators and document that methodology. The working group concept brought a sense of stability into the project because it as a organization remained constant and left a documented trail.

6.2 System Availability Issues

Another obstacle faced by the project was the uncertainty over the platform that would be used for the final product. One of the goals of the project is to be used as a console assistant. The system is to reside on a Mission Control Center Upgrade (MCCU) workstation. Unfortunately, the expert system project and the MCCU project have been developed in parallel. During the evolution of MCCU the final platform went from MASSCOMP to SUN back to MASSCOMP, from a two MIP's to an eight MIP's machine and from a non-color to a

color graphics terminals. Through it all we tried to keep the system as portable as possible and tried to get our requirements into the MCCU project. As a result of our philosophy of portability, the project did end up with some throw away code. The interface was done in four different versions; Curses, MC-Windows, SunView, and X-Windows versions. Even at this time, the MCCU's lack of robustness has caused the expert system setbacks in gaining controller confident since 90% of our sim failures can be attributed to workstation problems. It is difficult to avoid this type of problem when working in a high tech environment, on what might be called the leading edge of technology. The best that the development team has to be alert to changes and try to influence any changes to the environment.

As for any system, data is the life blood of the system. There existed two data problems for the ONAV expert system, availability and access. As previously stated the only acceptable data source, due to the size of the problem environment, is from simulations and flights. This proved to be a constraint on our development from the start. The main problem, however, was in getting access to the data required. The system requires telemetry data from the orbiter, for onboard systems and state vector evaluations, and data from the MOC for output of high speed ground filter data. Several sources existed from which telemetry data could be obtained. GDR, however, was the only source for the ground filter. A longer delay in achieving access to GDR data than anticipated occurred due to MOC integrity concerns. This caused an even larger delay between prototype development and rulebase verification. Possibly if we had tried to get a better understanding of the system's data requirements closer to the start of development; we may have been able to piece together a set of nominal testcases from some analysis tools. This would have given us a leg up on verification but not solved the whole problem. Also from our experience with verification the system's data requirements can expand through this process. This one factor stole the most momentum from the project and was one of the most frustrating experiences to endure. The system was all dressed up with no place to go.

6.3 CLIPS Development Issues

The C-language Integrated Production system (CLIPS) expert system shell was used in the development of the expert system. The ONAV project received one of first beta copies of CLIPS (release 3.0) for the prototype of the ONAV project. Even in the beta software, CLIPS inference engine was very robust. However, as the expert system development continued, more rules were added to the system. The rule based interactions became more complex. CLIPS at this time did not provide adequate debugging methods to deal with large knowledge bases, but in fact the state of the art for expert systems in general was lacking in this area. This was a particular problem because the expert system has to be embedded with other modules, such as the real time data acquisition, and user interface modules which added another layer of complexity. For a long time, tracking down a typographical error was very difficult. Since then, the CLIPS development team has developed additional tools such as the cross-reference, style, and verification (CRSV) tool, CLIPS window interfaces environments and also added additional syntax to the pattern language to deal with some of the problems. Finally, research is still focusing on finding the appropriate methodologies to perform verification and validation on expert systems. Certainly, CLIPS will also evolve along with the technology.

6.4 User Community and Management Issues

One area of particular concern was the acceptance of the system by users and management. One criticism from management was that the system would erode controllers skills, that a reliance on the system would eventually cause destruction of the ONAV knowledge base. However, our response was if the certification process for a controller is adequate then there should not be any erosion in skills. Users were also reluctant to use the system because the system did represent change and would require new training. Our approach was to bring in the users at the beginning, keep them involved and place the interface design in their control. Also all rulebase changes had to be approved by the expert. In these ways the user community became totally involved in the

development and responsibility for the project thus increasing acceptance by the users. The major factor in acceptance, however, is simply time. Time for the flight controllers to use the system and watch as the system's performance proves itself worthy of confidence. Time for management to accept that the system will not be death of flight controllers as we know them. Time for the system to become fully integrated into console operations.

7. CONCLUSION

To arrive at this point in time, where users are beginning to use the system, where it is actually being of benefit to operators, has been, as chronicled in the preceding passages, a somewhat tortuous path. In hindsight, a critical element in the development of the project was the length of time between rulebase development and prototype demonstrations. The length of time for our project proved to be much longer than the ideal, causing us severe problems but, ultimately, not disastrous ones. One important lesson was to have the assurance of the availability of experts and a first cut of data before starting rulebase development. Without this, a rapid development of a realistic prototype proved impossible, which prevented the users from actually using the system. Time on the system and with the system, by the users, as we have experienced is crucial for the development of the system. The working group concept should also have been used from the start of the project to facilitate documentation and provide stability for the system.

Time, it turned out was an enemy of this project. Too much was spent waiting; waiting for experts, waiting for workstations and waiting for data. As previously concluded, prototyping quickly, and keeping everyone involved is critical. Luckily, however, time will come to our rescue. We have spent a large amount of time on certification. This time has only increased the confidence of the controllers. And the more time spent in console operations, the greater the acceptance.

References:

- [1] Giarratano, J., *The CLIPS User's Guide*. NASA Document, June 1988

- [2] Riley, G., *CLIPS Architecture Manual*. NASA Document, JSC-23047, June, 1988
- [3] *Guidelines and System Requirement for the Onboard Navigation Console Expert /Trainer System*. NASA Document, JSC-22433, June 1986
- [4] Liebowitz, Jay, *Expert Systems with Applications Volume 1* November 1990
- [5] AAAI-90 Workshop on *Knowledge Based System Verification, Validation and Testing* Boston, Massachusetts July, 1990
- [6] *Knowledge Requirements for the Onboard Navigation (ONAV) Console Expert/Trainer System: Entry Phase*. NASA Document, JSC-22657, September, 1988
- [7] Culbert, Chris, *Approaches to the Verification of Rule-Based Expert Systems*, SOAR '87 August, 1987

XRTD: AN X-WINDOWS BASED, REAL-TIME RADIOMETRIC DISPLAY AND ANALYSIS SYSTEM

N 9 4 - 2 3 9 2 2

Vincent M. Pollmeier

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

ABSTRACT

XRTD is a graphical user interface (GUI) based tool for monitoring real time radiometric spacecraft data. The tool is designed to allow the navigation analyst to both view and analyze the characteristics of Doppler and ranging data. This capability is critical, if ground personnel wish to verify the correct performance of ongoing maneuvers. The raw tracking data is transferred from Deep Space Network (DSN) computers to a local workstation, where the predicted value for the observable is subtracted from the actual observed value to create a residual. The tool then allows the navigation analyst to rescale and replot the data using simple GUI techniques. The navigator may then perform a number of data analysis and modeling techniques on the resulting residuals to allow for the real time characterization of spacecraft events. These techniques include the modeling of maneuvers, the compression and differencing of data, and Fast Fourier transforms of the data. This tool has shortened the amount of time required for initial characterization of spacecraft maneuvers from several hours to a few minutes.

Key Words: Aerospace, space, navigation, operations, GUI, XWindows

1. INTRODUCTION

The display and analysis of radiometric tracking data is, and has always been, fundamental to the navigation of interplanetary missions. Traditionally, these two functions have been somewhat bifurcated. Display usually meant generation of plots of data residuals (the difference between the value of an observation and the expected value of that observation from the numerical models) to look for general trends in data priori to its analysis or to assure that no

gross abnormalities remained in the data after analysis. Analysis traditionally consisted of number-crunching on large main-frame computers and the output was ordinarily tabular text.

The workstation revolution, providing powerful compact computers equipped with high resolution graphics displays and high quality output devices, as well as the possibility of mouse driven operation of software, resulted in a new type of navigation tool, the graphical-analysis program. The computation speed of the workstations coupled with the growing prevalence of high speed data networks (such as those based on TCP/IP) further allows the graphical analysis tools to be connected to the real-time flow of data. This creates capabilities not previously possible and enhances previously existing ones to previously unthought-of levels.

1.1 Background

The basic radiometric observations received from spacecraft are Doppler and range data. Doppler data is simply the difference between the frequency of the radio signal received by the tracking station as compared to the frequency of the signal that was originally broadcast. This signal may have been broadcast by the receiving station and relayed by the spacecraft (2 way Doppler), transmitted by a second ground station and relayed by the spacecraft (3 way Doppler), or originated by the transmitter on the spacecraft itself (1 way Doppler). Range data can simply be thought of as the time delay between when a specified signal is broadcast from the originating ground station to when it is received back on the Earth. Like Doppler data, range data can be either 2 way or 3 way depending on if the transmitting station is the receiving station. (1 way range data is not used in interplanetary navigation.) Conversion of the received radiometric signal to data residuals

involves both the removal from the received signal of calibration values which depend on the ground station configuration and conversion of the predicted position and velocity of the spacecraft to an expected value of either Doppler shift or signal delay. In the case of Doppler data the received frequency of the data is subtracted from the transmitted frequency at a time which is twice the light time to the spacecraft before the receipt time. The primary problems with range data are applying the calibration for the delay induced by the receiving and transmitting stations and calculation of the ambiguity of the range delay. The ambiguity is such that for an observed signal delay, d , the actual delay is

$$nk + d \quad [\text{Eq. A}]$$

where k is a constant which depends on the ground station configuration and n , the range ambiguity, is a positive integer. Fortunately, k is generally large enough and *a priori* knowledge of the Earth relative range good enough that calculation of n is possible.

The primary purpose of a real-time graphical analysis tool is to perform real-time data validation and real-time spacecraft analysis. The first allows navigators to assess the effect of spacecraft and ground station configuration changes on data quality or availability. For example, this allows the determination of transmitter power to allow successful acquisition of data without requiring many passes of data to slowly determine the minimum value. The second purpose allows analysts to examine spacecraft activities as soon as the effects are visible at the Earth. These activities can include maneuvers or anything which changes the physical state of the spacecraft.

1.2 Precursor methods

In the past other methods have been used for near real-time data validation and analysis of spacecraft activities. The most common of these are referred to as real-time pseudo-residual displays and near real-time residual plots.

The first of these is a display capability provided by the DSN. The DSN calculates the expected received frequency received at a ground station based on a predicted spacecraft state file and based on predicted values of the ground station

configuration. When the actual signal is received, this predicted received frequency is differenced from the actual received frequency to calculate a pseudo-residual. These pseudo-residuals are then sent to low resolution displays which are in the possession of the flight team analysts. The drawbacks of the system are that a long lead time is required for the generation of these predicted values. If the ground station configuration at the receiver or transmitter is changed between the time the predicted values are generated and the actual observation, then the pseudo-residuals may be grossly wrong. Additionally, the fairly low resolution of the displays makes the determination of precise values difficult. No way to import the information from the displays into any analysis package exists. Finally, pseudo residual displays are not available for range data.

The second method for performing data validation tasks and for analyzing spacecraft activities was use of a near real-time residual plot. In general, this system was only used for analysis of maneuvers. This system involved using a high resolution plot of data generated by the analysts who first receive the data. The normal function of this group is to examine the data for gross calibration errors and if possible repair them and to remove obviously bad data points. However, this group can produce higher resolution plots of true residuals. Using these plots the navigation analysts then attempted to, by eye, place curves (usually linear) through the data and from the curves calculate the value of a maneuver. This system had the obvious drawback of being only as precise as the ability of the individual analyst to place the line through the data. For a typical spacecraft maneuver this could only be on the order of 5%. Additionally, the generation of these high resolution plots generally took in excess of one hour.

The desire to combine the true residuals and the high resolution capability of the near real-time residual plots and the real-time nature of the pseudo-residuals with a tool which removed the human element in placing curves through the data led directly to the design of the XRTD system.

2. SYSTEM DEVELOPMENT

Several basic capabilities are required of any real-time navigation graphical analysis tool. Access to real-time data must be had, calculation of residuals from the basic tracking parameters must be achieved, and then the display an analysis of the residuals must be accomplished.

The difficulty of combining all three of these functions into a single program as well as a desire to as much as possible utilize already existing software tools resulted in a decision to split the three function into four pieces of software. The first two transfer real-time data to the analyst's workstation, the third generates the residuals, and fourth is the actual graphical analysis software. The advantage of this system is that all three functions can be developed and tested separately. Additionally, much prior work has been done in the generation of residuals that could be drawn upon.

2.1 Data flow structure development

For simplicity, the decision was made to not attempt to develop interfaces directly with the data flowing from the DSN stations, but rather, to tap into the flow of data after it had already been collected at one point. The advantage of this is obvious, however the primary disadvantage is that the computer where all the data is brought together did not at the time of the software development have direct access to any analysts machine. However, a secure interface machine did exist and the decision was made to utilize it to provide the data flow. Two processes facilitate the data flow. The first resides on the intermediary microVAX. It tests for the existence of new data on the primary data collection machine and if it exists, then transfers the data from the data collection machine to the navigation workstation. The second simply receives the data on the navigation workstations and updates the data file on this machine. As this file is updated, the third piece of software generates the residuals and corresponding timetags. These are then output as an updated ASCII text file which can be read by the graphical analysis software. A block diagram of this data flow can be seen in Figure 1. The two tools which transfer the data are named

sprnet and *sprd*. The residual generation tool is called *rtd* and the graphical analysis tool is *Xrtd*

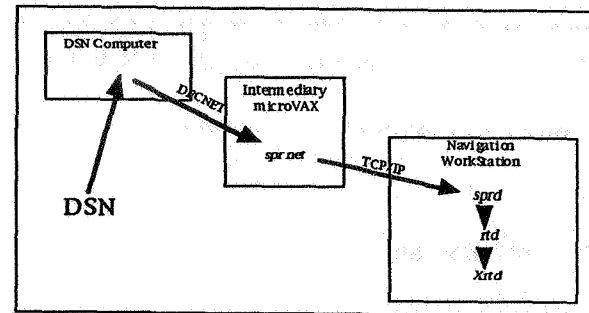


Figure 1: Block Diagram of Data Flow

2.2 Analysis tool development

One of the major advantages of the system described in the previous section is that the development of the graphical display and analysis tool is decoupled from the data acquisition and calculation portion of the problem. Analysis of commercially available graphical analysis tools indicated that none met the unique needs of the real-time display problem. Consequently, the decision was made to develop an independent tool to perform this analysis. The tool was designed using the X-Window System (commonly called Xwindows) developed by the Massachusetts Institute of Technology (Ref 1.). This windowing system provides the advantages of being portable and non-hardware vendor specific. Additionally, it provides basic capabilities from which any graphical user interface (GUI) can be built. The mode in which *Xrtd* was developed and designed to be used in integrates an Xwindows based system with the OSF/Motif GUI developed by the Open Software Foundation (Ref. 2). These two choices maximize the software portability.

The design philosophy was to provide a series of basic capabilities for displaying data such as rescaling the data plots and changing the time span of the plots which would not necessarily require use of keyboard inputs. Consequently, the plot span can be controlled via slider bars and the plot can be rescaled simply by selecting a rectangular portion of a plot with the mouse. The plot will be rescaled to display that portion on the whole plot.

Additionally, analysis of the data is undertaken by creating new plots which are described as "children" of the plot from which they originated. New "children" are spawned simply by accessing a command on a pull down menu. The "child" plots may in turn be treated as a "parent" and spawn more "children". Consequently, complicated adjustments and analysis may be performed on data by coupling simple processes together in series. A single "parent" plot can have multiple "children", thus allowing the on screen comparison of differing data analysis strategies. A graphical display of all plots is maintained to prevent the user from becoming confused and the user can access any plot by selecting its icon from the plot-tree. A number of "child" plots as well as the plot summary chart and the pull down menu which generates "child" plots can be seen in Figure 2.

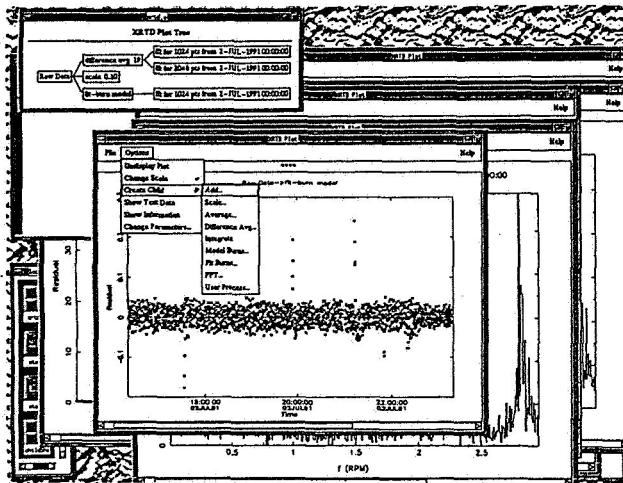


Figure 2: Xrtd windows with plot tree display

3. SYSTEM OPERATION

The entire *Xrtd* system is brought into operation when all four pieces of software are running. In practical application, the two pieces of software which transfer the raw tracking data run continuously. Only the latter two parts of the chain are generally started and stopped by the user. Additionally, the final part of the system, the *Xrtd* program itself, does not require real-time data, but can be run on old or simulated data. The *rtd* program is operated by giving it an input file which identifies the beginning time of data residuals to be calculated as well as the type (1 way Doppler, 2 way Doppler, 3 way Doppler, or range), and the desired tracking

station to be monitored. This then outputs the standard ASCII file which can be read by *Xrtd*.

3.1 Xrtd description

As could be seen in Figure 2, *Xrtd* has many capabilities. A brief discussion of the major ones will be given here.

3.1.1 File menu

The file menu has two entries, **Print Plot**, and **Exit**. **Print Plot** generates a PostScript image of the file which is then dumped to a PostScript compatible laserprinter. **Exit** exits the *Xrtd* program. A dialog box allows the user to abort the exit if desired.

3.1.2 Options menu

The **Options** menu is the primary menu of the *Xrtd* program. It allows you to perform most of the capabilities of the *Xrtd* program and has six entries.

3.1.2.1 Undisplay Plot

This command causes the current plot to cease to exist. This command is not accessible from the first plot.

3.1.2.2 Change Scale

This menu item allows the user to rescale the plot without using the mouse. This is to allow the user to scale the plot larger than that which is currently displayed.

3.1.2.2.1 Max Scale

This command changes the ordinate axis to include all data.

3.1.2.2.2 MAD Scale

This scaling changes the ordinate axis by using an algorithm designed to result in the best looking scale. This scale will not include isolated points which are grossly offscale.

3.1.2.2.3 Show All Data

This option changes both the ordinate and abscissa axes to include all data on scale.

3.1.2.2.4 Manual Scale

This allows the user to input exact scale values to be used. Both the ordinate and the abscissa may be input.

3.1.2.3 Create Child

The **Create Child** sub-menu is the route by which the user spawns new plots and analyzes the data.

3.1.2.3.1 Add

This simply adds a constant bias to all residuals

3.1.2.3.2 Scale

This input simply scales all residuals by an input scaling factor.

3.1.2.3.3 Average

This allows the user to perform a forward looking moving average of the data. The number of points to average is a user input.

3.1.2.3.4 Difference Average

This process calculates the difference between a given point and a centered average around the point. The number of points to use in the centered average is a user input. This process acts to filter out long period signatures in the data and to leave short period signatures present.

3.1.2.3.5 Integrate

This process numerically integrates the residuals. This serves to convert Doppler residuals into the equivalent phase residual. This process often allows the user to see data characteristics that are too small to be discerned in the unintegrated data.

3.1.2.3.6 Model Burns

This is a primary maneuver modeling tool. The user inputs the begin time, duration, initial value of residual at the beginning of the burn and final change in residual for any maneuvers. This capability allows the user to put in

maneuvers that have not yet occurred and assess very accurately in real-time the relative performance of the actual maneuver. If the maneuver has already occurred, the user can input all of the values by outlining the maneuver with the mouse. This is demonstrated in Figure 3.

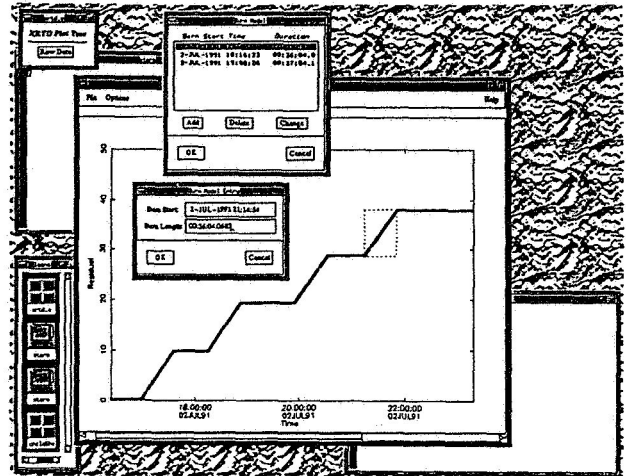


Figure 3: Use of mouse to input burn information

3.1.2.3.7 Fit Burns

This tool is similar to **Model Burns**, but rather than inputting the expected value for the maneuver. The software attempts to fit polynomials through the data. One polynomial is fit from the beginning of the data arc to the beginning of the user defined maneuver span, one is fit through the maneuver, and one is fit from the end of the maneuver to the end of the data arc or to the next maneuver. As with the **Model Burns** menu item, the initial time and duration can be input using the mouse rather than the keyboard.

3.1.2.3.8 FFT

This process will perform a Fast Fourier transform on the residuals. The beginning time for the transform and the number of points to be used are user inputs. The Fast Fourier Transform algorithm used requires the number of points used be a power of two. If not as many points exist after the beginning point as are requested, the plot will remain blank until enough points are available.

3.1.2.3.9 User Process

This item allows the user to input the name of a user written, compiled, and linked program. The program allows the user to create any additional processes that are needed. The program reads in the timetag and residual from standard input and outputs the new timetag and residual to standard output.

3.1.2.4 Show Text Data

The process shows a tabular listing of all the data displayed in the plot. The table is in three column format. The columns are point number, timetag (in Julian date), and residual.

3.1.2.5 Show Information

This shows the results of the **Fit Burns** process. The output are polynomial coefficients and the time spans to which they apply.

3.1.2.6 Change Parameters

This item allows the user to change the parameters of an already created window.

4. CONCLUSION

The *Xrtd* system has been utilized in its prototype form for two years. The system has performed wonderfully. Accurate determination of apparent maneuver performance has been achieved in only minutes compared to hours using previous systems. In addition, the Galileo Navigation Team has shown that the Fast Fourier Transform capability allows for rapid determination of spin rate and wobble amplitude. *Xrtd* has been used for the initial examination of the effectiveness of various attempts to free the Galileo High Gain Antenna. The system has regularly been used to determine the validity of data and to access in real-time the utility of various ground station configurations.

The system however still has some drawbacks. The data flow interface was generated to fit networks existing at the time of its inception. Improved networking ability may provide the opportunity to improve overall performance by changing the data flow path. Finally, like all systems which rely only on Doppler and range

data., all information obtained is only valid in the line of site between the receiving station and the spacecraft. To gain direct insight into the overall behavior, spacecraft telemetry data should be incorporated into the system. Currently, there is no capability for this. But it is hoped that future improvements to networking and data access will allow for the removal of these limitations.

5. ACKNOWLEDGEMENTS

The *Xrtd* system was originally developed as a Flight Project Support Office (FPSO) Advanced Technology Program proposal in 1990. Special acknowledgement must go to Earl Maize who led that original development as well as to those who contributed so much to the software used, Tim McElrath, Erik Slimko, and Peter Scott.

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

Reference herein to any specific commercial product, process, or service by trade name, trademark, manufacturer, or otherwise, does not constitute or imply its endorsement by the United States government or the Jet Propulsion Laboratory, California Institute of Technology.

6. REFERENCES

1. Quercia, V., and O'Reilly, T., 1991, *X Window System User's Guide, OSF/Motif Edition*, Sebastapol, California: O'Reilly and Associates, Inc.
2. Open Software Foundation, 1991, *OSF/Motif Style Guide, Revision 1.1*, Englewood Cliffs, New Jersey, Prentice Hall.

KALMAN FILTERING

N 9 4 - 2 3 9 2 3

APPLIED TO REAL-TIME MONITORING OF APOGEE MANEUVERS

Frédéric de BOER
MATRA MARCONI SPACE
Toulouse, France

Christian BARBIER
ALCATEL ESPACE
Toulouse, France

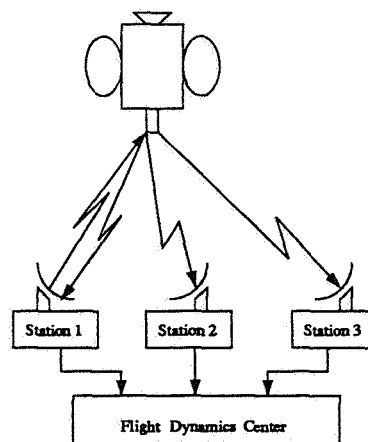
ABSTRACT

Part of the Space Mathematics Division in CNES, the Flight Dynamics Center provides attitude and orbit determinations and maneuvers during the Launch and Early Operation Phase (LEOP) of geostationary satellites.

Orbit determination is based on a Kalman filter method ; when the 2 GHz CNES/NASA network is used, Doppler measurements are available and allow orbit determination during the apogee maneuvers. This method was used for TELE-X and TDF 2 LEOP (3-axis controlled satellites) and also for TELECOM 2 and HISPASAT (spun satellites) : it enables us to follow the evolution of the maneuver and gives out a quite accurate estimation of the reached orbit.

In this paper, we briefly describe the dynamic models of the orbit evolution in both cases, "3-axis" and "inertial" thrust. Then, we present the results obtained for each case. Afterwards, we present some cases to show the robustness of the filter.

Key words : Kalman, real-time, apogee maneuvers.



Three stations configuration

1. REAL-TIME MONITORING PRINCIPLES

The principles of real-time orbit determination during apogee maneuvers have first been presented at 1985's CNES Congress by Bruno Belon and at 1992's IAF Congress by Didier Breton (Refs. 1-2).

The orbit determination is based on a Kalman filter method. The six orbital elements and the components of the thrust are estimated whenever a measurement is received. The measurements used are Doppler measurements.

The 2 GHz CNES-NASA network authorizes continuous Doppler measurements. The best results are obtained when the apogee maneuver occurs with TM/TC visibility of three different stations. In this case, the prime station provides 2-way measurements and composite measurements with the two other stations.

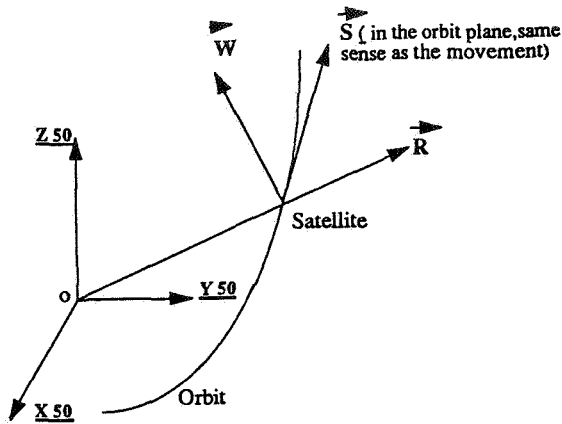
1.1 State Vector and Dynamics Model for a Three-axis Thrust

Kalman filtering is applied to the estimation of the following nine components state vector :

$ \begin{aligned} l &= v + w + \Omega \\ a &= a \\ k &= e \cos(w + \Omega) \\ h &= e \sin(w + \Omega) \\ q &= \tan(i/2) \cos \Omega \\ p &= \tan(i/2) \sin \Omega \end{aligned} $	equinoxial orbital parameters
$ \begin{aligned} \gamma_r \\ \gamma_s \\ \gamma_w \end{aligned} $	components of $\vec{F}/m$ ($\vec{F}$: thrust, m : mass) in the local orbital frame

Note : (a, e, i, w, Ω, v) are the keplerian parameters with v , the true anomaly.

The local orbital frame is defined as follows :



The three last components of the state vector are chosen to be adapted to "3-axis" satellites where the thrust has a fixed direction in the local orbital frame.

The evolution of the orbital elements is given by the Gauss equations :

$$\begin{bmatrix} \frac{dl}{dt} \\ \frac{da}{dt} \\ \frac{dk}{dt} \\ \frac{dh}{dt} \\ \frac{dq}{dt} \\ \frac{dp}{dt} \end{bmatrix} = \begin{bmatrix} G_{00} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 & 0 & G_{13} \\ G_{21} & G_{22} & 0 \\ G_{31} & G_{32} & G_{33} \\ G_{41} & G_{42} & G_{43} \\ 0 & 0 & G_{53} \\ 0 & 0 & G_{63} \end{bmatrix} \begin{bmatrix} R \\ S \\ W \end{bmatrix}$$

$[G_{00}, G_{13}, G_{21}, G_{22}, G_{31}, G_{32}, G_{33}, G_{41}, G_{42}, G_{43}, G_{53}, G_{63}]$ are called gaussian coefficients. G_{00} corresponds to the keplerian movement. R, S, W are the components of the perturbing acceleration in the local orbital frame (considered as an inertial frame).

And the evolution of the thrust acceleration is given by the three equations :

$$\begin{cases} d\gamma_R/dt = q/m \gamma_R \\ d\gamma_S/dt = q/m \gamma_S \\ d\gamma_W/dt = q/m \gamma_W \end{cases} \quad \left| \quad q : \text{flowrate} \right.$$

This modelisation is only valid for a 3-axis thrust because it supposes that the components of the thrust are constant in the local orbital frame.

The nine-equation system of the dynamic model is integrated during the prediction phase of the Kalman filtering ; for this integration we need to compute R, S, W , the components of the perturbing acceleration due to the thrust ($\gamma_R, \gamma_S, \gamma_W$) and the acceleration caused by the J_2 term of the terrestrial potential (every other perturbing factor is considered to be less important than the modeling error of the thrust).

1.2 State Vector and Dynamics Model for an "Inertial" Thrust

In the case of spun satellites during transfer (TELECOM 2, HISPASAT), the thrust has a fixed direction in an inertial frame. The dynamic model used for 3-axis transfer satellites, which supposed a fixed thrust direction in the local orbital frame, has to be adapted.

The state vector becomes :

$$\begin{bmatrix} l \\ a \\ k \\ h \\ q \\ p \\ \gamma_x \\ \gamma_y \\ \gamma_z \end{bmatrix} \quad \left| \quad \begin{array}{l} \text{equinoxial parameters} \\ \text{components of acceleration} \\ \text{due to the thrust in } \Gamma_{50} \text{ inertial} \\ \text{frame : F/m} \end{array} \right.$$

With this choice, the dynamic model has a similar formulation to the 3-axis case.

$$\begin{bmatrix} \frac{dl}{dt} \\ \frac{da}{dt} \\ \frac{dk}{dt} \\ \frac{dh}{dt} \\ \frac{dq}{dt} \\ \frac{dp}{dt} \end{bmatrix} = \begin{bmatrix} G_{00} \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} G'_{11} & G'_{12} & G'_{13} \\ G'_{21} & G'_{22} & G'_{23} \\ G'_{31} & G'_{32} & G'_{33} \\ G'_{41} & G'_{42} & G'_{43} \\ G'_{51} & G'_{52} & G'_{53} \\ G'_{61} & G'_{62} & G'_{63} \end{bmatrix} \begin{bmatrix} X \\ Y \\ Z \end{bmatrix}$$

$$d\gamma_X/dt = q/m \gamma_X$$

$$d\gamma_Y/dt = q/m \gamma_Y$$

$$d\gamma_Z/dt = q/m \gamma_Z$$

X, Y, Z are the components of the perturbing acceleration (thrust + J_2 term of the terrestrial potential in Γ_{50} frame).

The new Gauss coefficients are obtained from the previous ones by the relation :

$$|G'_{ij}| = |G_{ij}| \cdot (R)$$

R is the rotation matrix from the local frame to the inertial frame Γ_{50} .

As for the "3-axis" satellites, the components of the perturbing force (X, Y, Z) take into account the components due to the thrust ($\gamma_X, \gamma_Y, \gamma_Z$) and the acceleration caused by the J2 term.

1.3 Measurement Model

When a frequency f_t is transmitted to the satellite and received back on the earth, the received frequency is $f_r = k f_t (1 - 2 V/c)$ (k is a satellite characteristic). The measurement of f_r , which lasts a certain time T_c (counting time) provides V , the station-satellite removing velocity. In case of composite measurements (transmitting station different from receiving station), the measured velocity is the average of the removing velocities from the two stations.

In case of two-way Doppler measurements, the theoretical measurements V_{th} is given by :

$$TC \cdot V_{th} = D(t + TC) - D(t)$$

$D(t)$ is the satellite-station distance at the t date, $D(t + TC)$ is the predicted satellite-station distance at $t + TC$. These two distances are computed as a function of the state vector. The measurements are taken into account in the correction phase of the filter.

2. APPLICATIONS

For practical applications during apogee maneuvers, the correction phase of the filter is only active from the beginning of the firing on. Before that, the orbit estimated by the software is only an extrapolation of the initial orbit (prediction phase of the filter). The measurements are used to correct the orbit only when the firing has begun. This appears clearly in the displays of the residuals shown in this paper immediately after the beginning of the firing, the correction makes the residuals decrease.

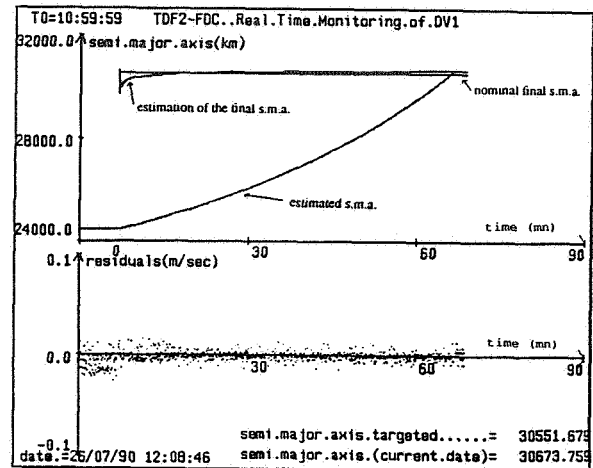
We will present now the results obtained in two cases :

- three-axis satellites,
- spun satellites.

2.1 Three-axis Satellite - TDF 2

The apogee maneuvers (first and third) during TDF2 LEOP occurred with the visibility of three 2 GHz CNES stations : Kourou (KRU), Aussaguel (AUS) and Hartebeesthoek (HBK).

On the graphic hereafter, the evolution of the semi-major axis and the residuals (after correction) during the firing are presented. Every time a new value of the semi-major axis is estimated, an estimation of the final semi-major axis is computed as a function of the current estimated orbit, the current estimated thrust and the remaining firing duration.



With regard to the first apogee maneuver, if we compare the real-time estimated orbit with the real orbit at the end of the firing, we notice that the results are very accurate. About one hundredth of degrees on w , Ω , M , one thousandth on i . For a , we reach a 5 km precision (to be compared with the difference between the real and the targeted orbit : 120 km).

The precision on the semi-major axis can be increased by following on the orbit restitution a few minutes after the end of the firing.

	Orbit at the beginning of the firing	Nominal orbit at the end of the firing	"Real" orbit at the end of the firing	Real-time estimated orbit at the end of the firing
Date	26/7/90 11h06'57.113"	26/7/90 12h06'41.114"	26/7/90 12h06'41.114"	26/7/90 12h06'41.114"
a (km)	24433.124	30551.677	30673.097	30668.092
e	0.7264	0.3820	0.3766	0.3768
i (deg.)	4.042	1.251	1.3747	1.3736
w (deg)	-180.255	179.014	178.639	178.667
Ω (deg)	90.020	90.448	90.407	90.380
M (deg)	162.538	192.352	193.129	193.119

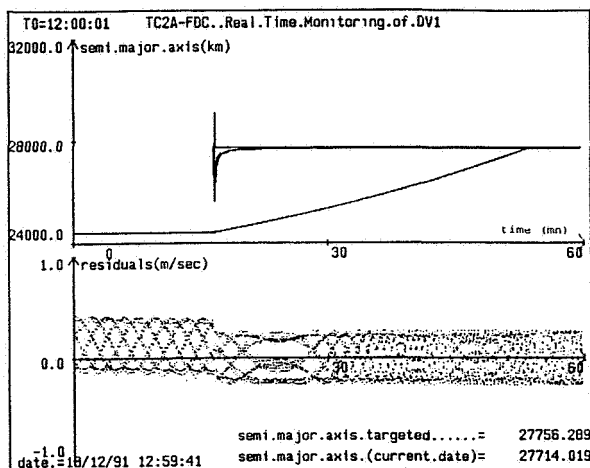
Nota : With regard to the third apogee maneuver, the precision on the semi-major axis obtained is around 10 km.

2.2 Spun Satellites : TELECOM 2, HISPASAT

2.2.1 TELECOM 2

Turning to TELECOM 2-A (first and third apogee maneuvers) and the first apogee maneuver of TELECOM 2-B, the firing occurred with the visibility of KRU, AUS and HBK CNES stations.

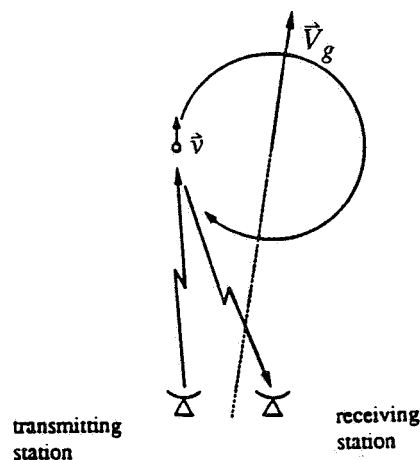
Concerning the second apogee maneuver of TELECOM 2-B, the real time monitoring occurred with the visibility of KRU, AUS CNES stations and Goldstone (GDS) NASA station. Hereafter, we present the results obtained for TELECOM 2-A first apogee maneuver, the other results are synthesized at the end of this chapter.



	Orbit at the beginning of the firing	Nominal orbit at the end of the firing	"Real" orbit at the end of the firing (computed 12 h later)	Real-time estimated orbit at the end of the firing
Date	18/12/91 12h16'36.864"	18/12/91 12h53'31.066"	18/12/91 12h53'31.066"	18/12/91 12h53'31.066"
a (km)	24415.89	27756.29	27719.67	27717.17
e	0.726	0.518	0.521	0.521
i (deg.)	3.983	1.914	1.959	1.9417
w (deg)	179.023	178.636	178.625	178.720
Ω (deg)	247.187	247.471	247.502	247.405
M (deg)	172.292	190.613	190.596	190.593

The precision on the estimated orbit is quite good (comparable with 3-axis satellites) though the fluctuations of the residuals are more important : 60 cm/s instead of 2 cm/s for 3-axis satellites.

The fluctuations of the residuals describe a periodic phenomenon which is caused by the rotation of the satellite. The Doppler measurements correspond to the velocity of S-band antenna with respect to the station. As the S-band antenna is not localized on the spin axis of the satellite, the measurement is the superposition of the periodic movement of the antenna with respect to the satellite gravity center and the movement of the satellite with respect to the station. The counting time lasts 2 seconds for CNES stations and the spin period is about 5 seconds. During some measurements, the velocity of the antenna will be added to the velocity of the satellite gravity center; during some others, it will be subtracted.



$\vec{V}_g$: satellite velocity in the station satellite direction,
 $\vec{v}$: S-band antenna velocity with respect to the spin axis,

The received frequency is :

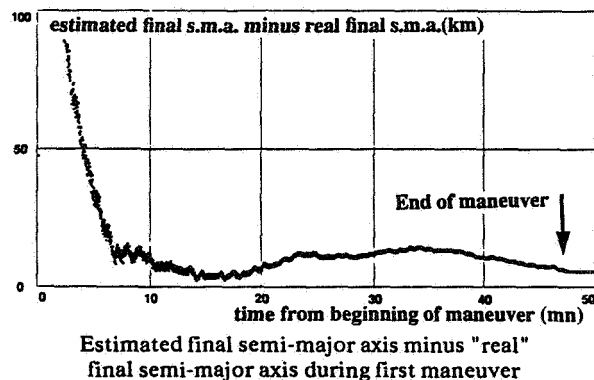
$$f_r = k f_e \cdot \left(1 - \frac{2}{c} \cdot \left(\vec{V}_g + \frac{\vec{v} \cdot \vec{V}_g}{\|\vec{V}_g\|} \right) \right)$$

2.2.2 HISPASAT

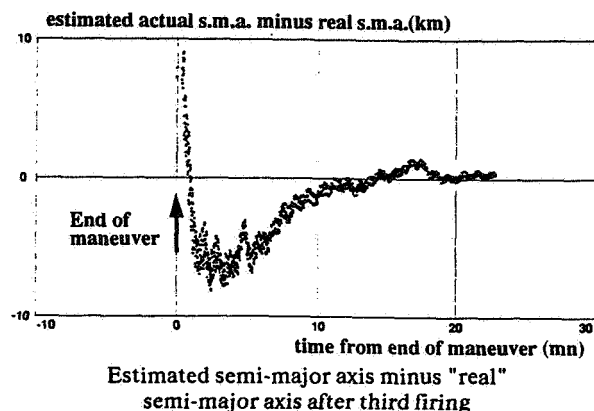
The first and the third apogee maneuvers of HISPASAT occurred with the visibility of HBK, AUS and KRU CNES stations. The precision obtained on semi-major axis is less than 5 km for the first maneuver.

	Orbit at the beginning of the firing	Nominal orbit at the end of the firing	"Real" orbit at the end of the firing	Real-time estimated orbit at the end of the firing
Date	12/09/92 11h54'21.914"	12/09/92 12h41'48.332"	12/09/92 12h41'48.332"	12/09/92 12h41'48.332"
a (km)	24347.74	29366.43	29284.81	29289.2
e	0.72954	0.43432	0.438334	0.438142
i (deg.)	6.966	2.643	2.725	2.673
w (deg)	178.989	178.305	178.192	177.919
Ω (deg)	151.431	150.306	150.461	150.739
M (deg)	173.332	197.819	197.829	197.824

The following figure shows the convergence of the estimation of the final semi-major axis during the maneuver. This convergence is reached about 10 minutes after the beginning of the firing and the deviation with respect to the "real" semi-major axis at the end of the maneuver is then less than 15 km.



The third apogee maneuver monitoring gave a precision on semi-major axis of about 25 km just at the end of the maneuver. This relatively bad precision could be explained by the fact that the estimation of the final semi-major axis has not totally converged before the end of the firing. It is often the case for third apogee maneuvers which are usually very short. However, about ten minutes after the end of the maneuver, the semi-major axis has converged to less than 2 km of its "real" value.



2.2.3 Synthesis of Results for Spun Satellites

Hereafter, we have summarized the results obtained in the case of TELECOM 2 and HISPASAT. We have compared the orbit computed a few hours after the maneuvers with the orbit estimated at the end of the firing.

The accuracy obtained for the semi-major axis is less than 10 km except for the second apogee maneuver of TELECOM 2-B. In that case, the elevation of the satellite seen from the station of Aussaguel was less than 5 degrees, the resulting bad quality of measurements could have contributed to this lack of precision. On the whole, the accuracy obtained on the other components of the orbit is quite good.

	Stations	Thrust duration	Da (km)	De	Di (deg)	Dw (deg)	D Ω (deg)	DM (deg)	DI (deg)
DV1 TC 2-A	KRU,AUS,HBK	2214.2	-1.79	$5 \cdot 10^{-4}$	$-9 \cdot 10^{-3}$	$8.3 \cdot 10^{-2}$	$-8.6 \cdot 10^{-2}$	$6 \cdot 10^{-3}$	-
DV3 TC 2-A	KRU,AUS,HBK	227.4	6.04	$-8.1 \cdot 10^{-5}$	$1.6 \cdot 10^{-3}$	-	-	-	$3 \cdot 10^{-3}$
DV1 TC 2-B	KRU,AUS,HBK	2270.8	-8.9	$4.87 \cdot 10^{-4}$	$7 \cdot 10^{-3}$	$7.1 \cdot 10^{-2}$	$-6.6 \cdot 10^{-2}$	$-3 \cdot 10^{-3}$	-
DV2 TC 2-B	KRU,AUS,GDS	3240.123	13.12	$-8.7 \cdot 10^{-5}$	$-3.67 \cdot 10^{-2}$	-	-	-	-0.1
DV1 HISP	HBK,KRU,AUS	2846.418	4.4	$-1.9 \cdot 10^{-4}$	$-5.2 \cdot 10^{-2}$	$-2.7 \cdot 10^{-1}$	$2.8 \cdot 10^{-1}$	$-5 \cdot 10^{-3}$	-
DV3 HISP	HBK,KRU,AUS	278.218	< 1.	$3 \cdot 10^{-5}$	$4.8 \cdot 10^{-2}$	-	-	-	$9 \cdot 10^{-3}$

Note : The values correspond to the orbit determined after a few minutes of convergence.

3. ROBUSTNESS OF REAL-TIME MONITORING

The software has shown a good behavior during all the apogee maneuvers and the orbital parameters obtained after the maneuvers were quite accurate. However, all these cases were nominal and no major problems were encountered. Furthermore, the only way to control the validity of the results is to compare them to an orbit determined after a few hours of ranging. A question that always arises at the end of the maneuver is: what confidence do we have in the orbit determined by the real-time monitoring?

Some simulations have therefore been performed to test the robustness of the software to high dispersions on the thrust, on the attitude of the satellite, as well as delays on the thrust duration or on the maneuver beginning time. TC 2-A and HISPASAT first apogee maneuvers were considered and the simulation has consisted in monitoring these cases with wrong initial data. The considered cases were an underperformance of 50 N on the thrust (10 % of the nominal value), a bias of 5 deg. on the right ascension or on the declination of the thrust direction, and errors of 10 seconds on the thrust duration and the beginning time. It has to be noticed that these values are far above the specifications on these parameters, and can be considered as degraded cases.

The results obtained are quite good for all cases. The estimation of the final semi-major axis during the maneuver rapidly converges to the value obtained in the nominal case except for the error on thrust duration: the theoretical remaining duration is in that case different from the nominal one. The final orbital parameters at the end of the maneuver obtained after convergence are also quite accurate because the deviation with respect to the parameters obtained by the nominal real-time monitoring are of a few hundredths of degrees on w and Ω , a few thousandths on the mean anomaly.

The worst result obtained on the inclination deviation is of 1.5 hundredth of degrees obtained for the bias on the attitude. The deviation on the semi-major axis is always less than 2 km. The results for HISPASAT are gathered in table 3.1.

4. CONCLUSION

Real-time monitoring is possible when continuous Doppler measurements from three different stations are available. It enables us to follow the progress of the orbital parameters during the maneuver for three-axis stabilized satellites as well as for spun stabilized satellites. It provides then a quite accurate first diagnosis of the orbit a few minutes after the maneuver. The precision reached on the final orbit is of a few kilometers on semi-major axis. Moreover, the software has shown its robustness with respect to degraded cases on thrust amplitude or direction, as well as errors on the thrust duration or beginning time.

Another point is the fact that, with this software, we are able, as soon as the firing is over, to compute the new predictions (station visibilities, eclipses,...), another apogee maneuvers strategy if necessary. Before, we had to wait many hours or to compute it with the targeted orbit.

REFERENCES

1. Belon, Bruno. Real-time determination during apogee maneuvers, CNES Congress, 1985.
2. Breton, Didier. Kalman filtering applied to real-time monitoring of apogee maneuvers. 43rd Congress of the International Aeronautical Federation, Washington, August 28-September 5, 1992.

Case	Da (km)	De	Di (deg)	Dw (deg)	D Ω (deg)	DM (deg)
50 N underperformance of the thrust	-0.8	$6 \cdot 10^{-5}$	$2 \cdot 10^{-3}$	$1 \cdot 10^{-2}$	$-2 \cdot 10^{-2}$	$1 \cdot 10^{-3}$
5 degrees bias on the initial thrust declination in Γ_{50} frame	-0.2	$1.2 \cdot 10^{-5}$	$1.5 \cdot 10^{-2}$	$2.5 \cdot 10^{-1}$	$2.5 \cdot 10^{-1}$	$< 1 \cdot 10^{-3}$
5 degrees bias on the initial thrust right ascension in Γ_{50} frame	-1.8	$1.4 \cdot 10^{-4}$	$3 \cdot 10^{-3}$	$3 \cdot 10^{-2}$	$-3 \cdot 10^{-2}$	$-6 \cdot 10^{-3}$
10 s delay on the beginning time	+1.3	$4.5 \cdot 10^{-5}$	$1 \cdot 10^{-3}$	$3 \cdot 10^{-2}$	$5 \cdot 10^{-2}$	$5 \cdot 10^{-3}$
Burn out 10 s earlier	-1.3	$2 \cdot 10^{-5}$	$3 \cdot 10^{-3}$	$1.6 \cdot 10^{-1}$	$1 \cdot 10^{-2}$	$2 \cdot 10^{-3}$

Table 3.1

Mars 94/96: The French Navigation Tasks

H.7

Frédéric Bonneau, Jacques Bernard
CNES, Toulouse, France
Damien Delobette
Sema-Group, Toulouse, France

N 9 4 - 2 3 9 2 4

1. Abstract

In Fall 1994, Russia will launch a spacecraft to Mars. France is involved in many scientific experiments which are on board the spacecraft, as PI or CI. Some days before Mars Orbit Insertion Maneuver, 2 small stations and 2 penetrators will be injected into an entry trajectory, and they will carry out during at least 6 months in situ analysis on Mars Surface.

Two years after, a second spacecraft will be launched. It will carry the French Balloon, and also a small rover.

The scientific data of these landers will be relayed to earth via the spacecraft. However, during the first 20 days of their mission, Mars Observer will be used. To this end, a Mars Balloon Relay will be used, which will receive the data from the landers, store them into the memory of the Mars Observer Camera.

The spacecraft will be also used to localize the landers, with the help of relative 1 way Doppler measurements. An international cooperation is set up for this process, including JPL, Russian ballistic centers (Babakine, Institute of Applied Mathematics, Moscow Flight Control Center), and CNES Toulouse.

Another task dedicated to the Space mathematics division of CNES is to support the french scientists to prepare their telecommands and to analyze their telemetry. This second part is integrated into the French Ground Segment created for the Mars94/96 Mission.

This paper will describe the method used in Cnes for the localization process, the support to the scientists and the links for the data exchange

Keywords

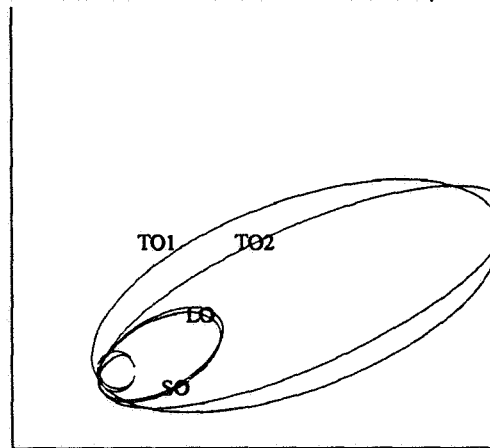
Mars Mission, Mars landers, lander localization, Mars 94 project

2. The Mars 94 Mission

The Mars 94 spacecraft will be launched by Proton at the end of October 1994. It will arrive at Mars early September 1995.

Some 5 to 7 days before arrival, 2 small stations and 2 penetrators will be released, two of them will land near the equator of Mars, the other 2 at about 40 degree north latitude.

Before being inserted onto its final scientific orbit, the spacecraft will be inserted into 3 intermediate orbits (see fig 1). It will remain on the high elliptical orbits during roughly 20 days, after which it will be inserted into an orbit of period $T_{sol}/2$ (T_{sol} is a Martian day). It is on this orbit that the spacecraft will mainly relay the scientific data of the landers. Therefore, during the



TO1: $T=72h$, $i=16\text{Deg}$, $H_p=500$
TO2: $T=72h$, $i=90\text{Deg}$, $H_p=300$
LO: $T = T_{sol}/2$, $i=90\text{Deg}$, $H_p=300$
SO: $T=12h$, $i=90\text{Deg}$, $H_p=300$

Fig 1: Mars 94 orbits

first 20 days of the martian mission, Mars Observer will be used to relay the data.

In Fig2, there is an example on a ground track of Mars Observer, during 3 orbits:

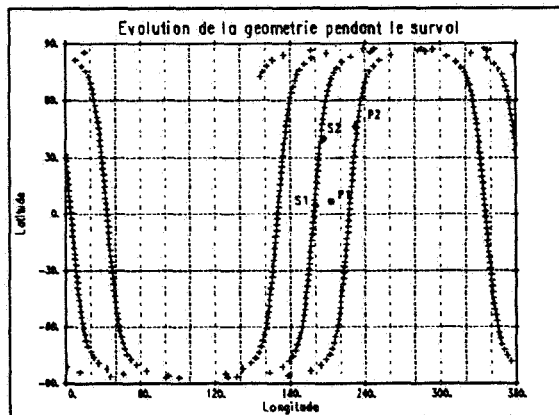


Fig 2: ground track of Mars Observer
S1 and S2 are the small stations,
P1 and P2 the penetrators

3. The localization process

3.1 The measurements:

The landers send to the spacecraft a signal with a nominal frequency f_0 .

This frequency is affected by a bias and a drift. Therefore, the real emitted frequency is:

$$f_r = f_0 + b + d \cdot (t - t_0) + \epsilon$$

The value of b and d is random for each pass, but constant over one pass. t_0 is the time of the beginning of the measurements.

The nominal frequency f_0 is 401.5275 Mhz.

According to specialists from Babakine, the nominal values of b and d are:

for the penetrators:

$$b = 10^{-5} \cdot f_0, d = 10^{-8} \cdot f_0 / 15'$$

for the small stations:

$$d = 2 \cdot 10^{-6} \cdot f_0, d = 10^{-8} \cdot f_0 / 15'$$

The random noise ϵ is $3 \cdot 10^{-9} \cdot f_0$, so roughly 1.2 Hz.

3.2 The software developed in CNES: FILON (Filtrage for Localization and Navigation)

We have developed in CNES a kalman filter, by upgrading the initial version used for the Phobos optical navigation. This software is a multi-purpose software of localization of objects on a surface of a planet with spacecraft relative measurements. It is currently

used for Mars 94/96, for the VAP project (the french rover studies), for the TAOS and S80T projects (mini earth satellite for localization of mobiles).

In the current version, the state vector has up to 20 components: 6 for the mobile, 7 for the spacecraft, and other for bias, drift, ionosphere, measurements error.

In the case of the Mars94 project, the only measurements which are processed by the software are relative measurements, so 1 way Doppler from the landers to the spacecraft. More precisely, we don't currently process any ground based measurements.

Therefore, the initial state vector for the spacecraft is computed either by JPL in the case of MO, or the Russian ballistic centers (IPM, TSOUP) in the case of Mars94.

For performance evaluation, the software is included into the following process (see fig 3)

Starting from exact position of the landers, and an exact orbit, the doppler measurements are simulated, and are perturbed by a gaussian noise.

In a second step, the location of the landers, the satellite orbit, the ancillary parameters are perturbed according to their assumed covariances.

In a third step, starting from those points, FILON is executed, and gives final values of those parameters.

In a final step, all those solutions are compared

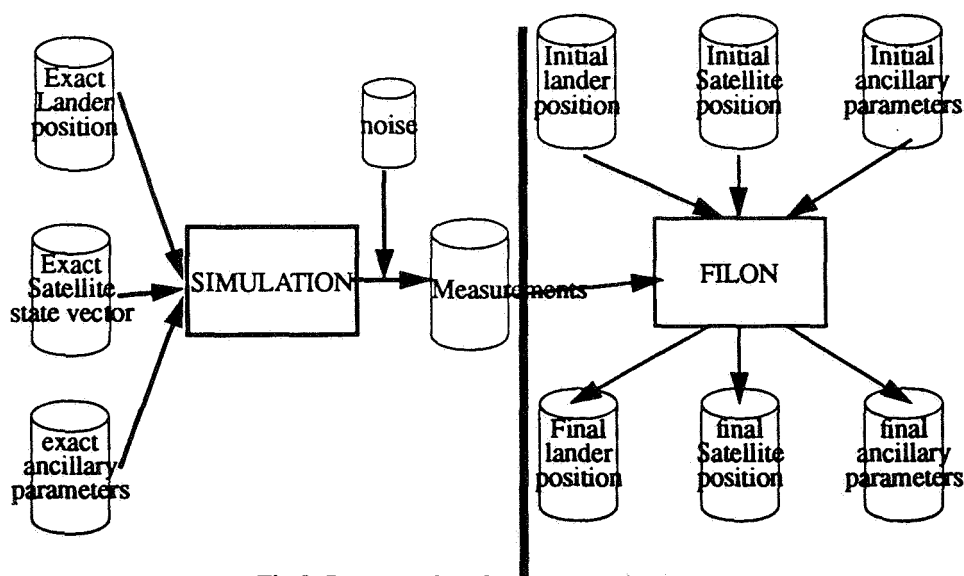


Fig 3: Process of performance evaluation

together with their formal covariances, with the help of utility programs, and graphical displays are used.

4. Results for the first localization process

For the first localization process, Mars Observer will be used as relay. The dispersions at landing will be large (some degrees in longitude and latitude), and so it will be the most difficult arc to process, as far as the convergence is concerned.

One important factor for the accuracy of the localization process is the accuracy of the spacecraft orbit. Therefore, we have processed different cases, depending on the elapse time between the initial conditions, and the time of the beginning of the measurements. We have considered 4 cases:

Solution T0: It's assumed that the result of the orbit determination process is given at the time of pericenter. Therefore, we deal with reconstructed errors. The value of these errors are issued from the Nav plan, and are:

Radial: 4.5 m Along track: 49 m Cross track: 240m

Solution T1: It's assumed that we need 1 day extrapolation from the last OD solution. Therefore, the errors at time of pericenter are:

Radial: 4.6m Along track: 120m Cross track: 240m

Solution T3: It's assumed that we need 3 day extrapolation from the last OD solution. Therefore, the errors at time of pericenter are:

Radial: 8.7m Along track: 1210m Cross track: 240m

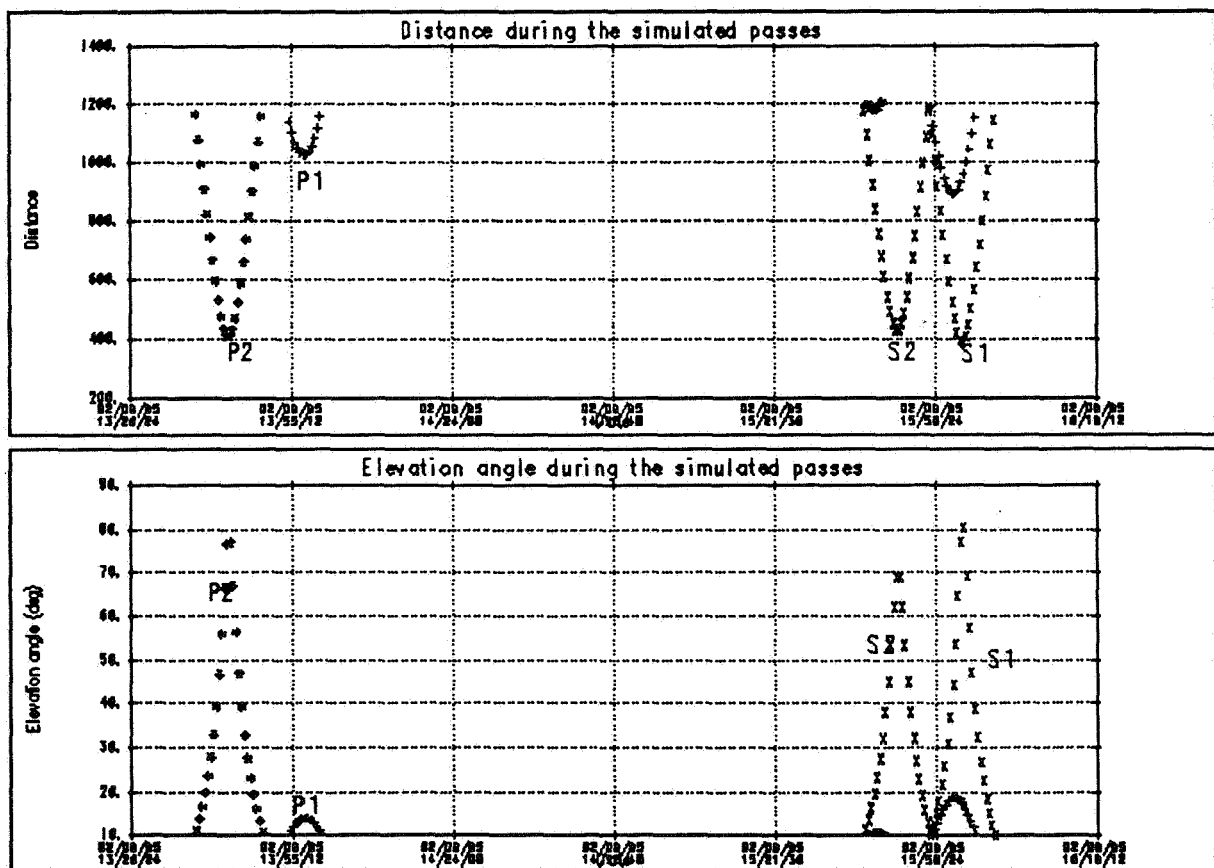


Fig 4: Characteristics of the passes for MO

Solution T7: It's assumed that we need 7 day extrapolation from the last OD solution. Therefore, the errors at time of pericenter are:

Radial: 38.m Along track: 7080m Cross track: 260m

According to the geometry indicated in figure 2, the geometry of the relay is shown in fig 4. We can see that, for the studied arcs, there are 2 visibility for each penetrator, one at each orbit. The small stations can be relayed just at the second MO orbit.

4.1 Results for penetrator P1:

Fig 5 shows the 1 sigma ellipses obtained with the 4 different cases

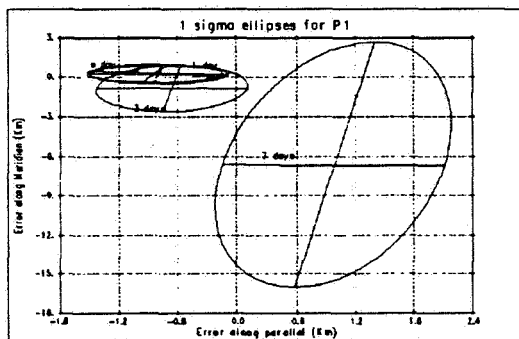


Fig 5: 1 sigma ellipses for P1

4.2 Results for penetrator P2

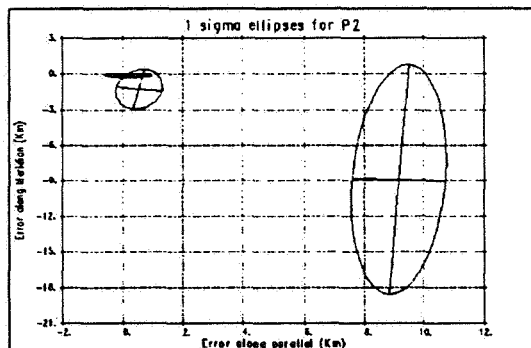


Fig 6: 1 sigma ellipses for P2

4.3 Results for small station S1

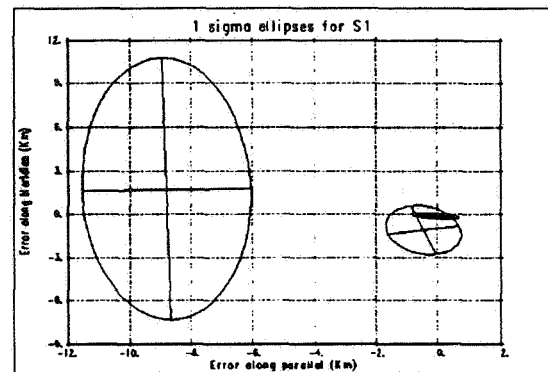


Fig 7: 1 sigma ellipses for S1

4.4 Results for small station S2

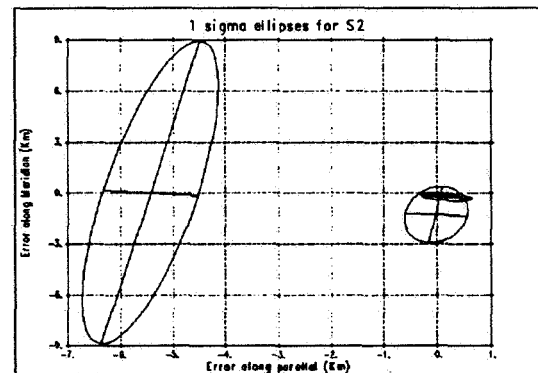


Fig 8: 1 sigma ellipses for S2

4.5 Analysis

If the extrapolation on the S/C state vector is made not longer than 3 days, the accuracy on the location of the landers is roughly 1 km. A 7 day extrapolation leads to a poor accuracy, and moreover the convergence is slow.

However, in any case, the accuracy is much better than the requirements for pointing the antennas toward the landing site.

5. Results with Mars94

As it was shown in the above paragraph, the accuracy on the lander position is strongly dependent of the accuracy on the orbit of the spacecraft itself.

For Mars 94, we don't have yet enough results on this

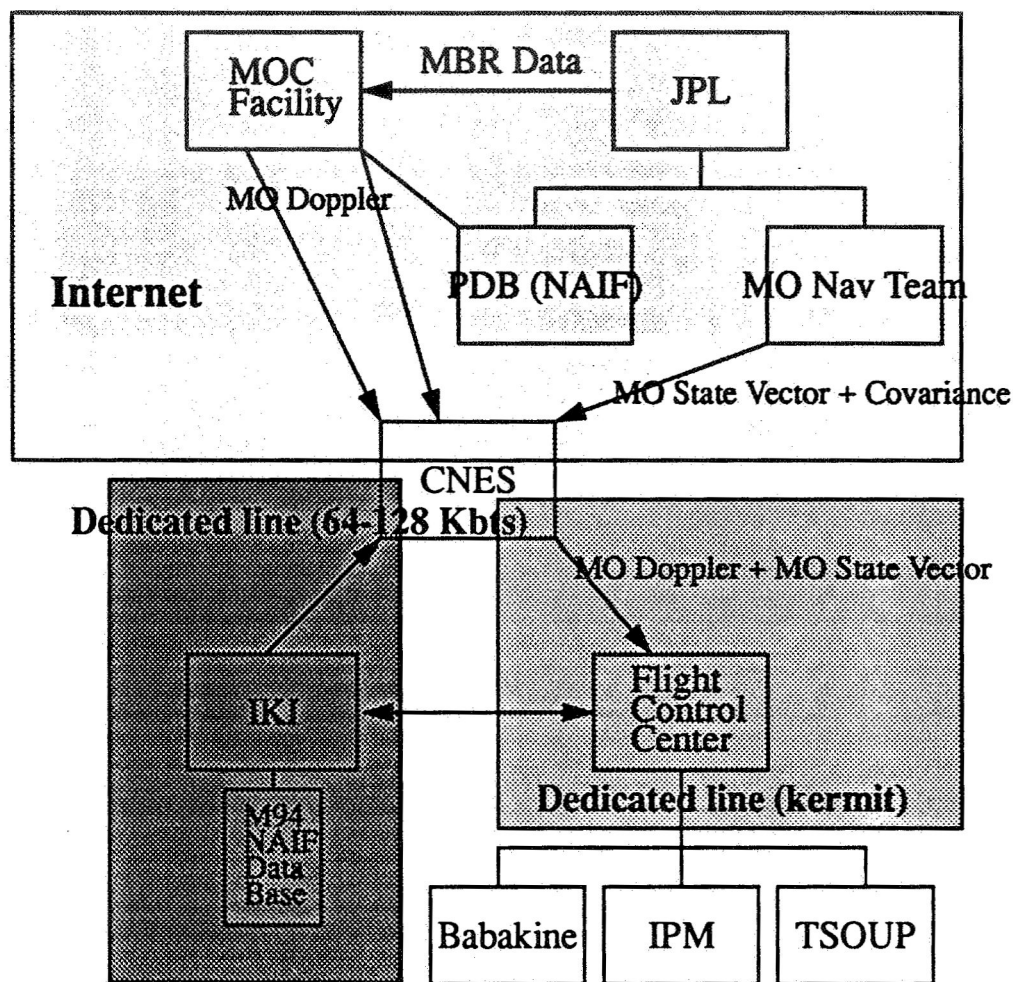


Fig 9: Proposed Data flow exchange for Navigation

accuracy to execute the localization process. However, we can assume that, at the time when M94 will be used as relay, the accuracy on the lander position will be better than the accuracy on the S/C orbit. Therefore, those landers may be used as reference beacon to improve the orbit.

6. The International cooperation:

The localization process of the small stations and penetrators will involve different partners in different countries:

USA:

MOC Facility, in San Diego, which will receive the

telemetry

JPL: MO navigation Team, and M94 localization Team

Russia:

Ballistic Centers: TSOUP, IPM, Babakine, which will constitute the Flight Control Center

IKI (Institute of Cosmic Research): Which will receive the telemetry relayed by M94.

France: CNES Toulouse

It has been approved that all information concerning the Spacecraft state vector will be sent using NAIF (Navigation Ancillary Information File) system, developed by JPL. The access to the NAIF system for MO will be made through the MOC Facility, with

connection to the Project Data Base (PDB).

As far as Mars94 is concerned, IKI is responsible of the NAIF data base.

The proposed data flow exchange is indicated in fig 9. The communication with the US will use commercial lines (Internet). With Russia, dedicated lines will be set up by CNES. With IKI, it will be a high baud rate line, using Intelsat Satellite. This line will mainly be used for real time receiving of the scientific telemetry. With the Flight Control Center, which will gather the 3 ballistic centers, it will be a 9600 B/s line, using Kermit.

The timing and responsibilities of the exchange is still under discussion between the 3 countries.

7. Support to the french scientists

Another part of the navigation tasks is to help the scientists to prepare their telecommands, and to analyze their results. This consists mainly into prediction of events, with a poor accuracy, for the telecommands. Those predictions will be updated and recomputed with a better accuracy, for the analysis of the results. For this task, we will use the NAIF system. The main scientific experiments on board the Mars 94 S/C or landers are:

SPICAM: (solar and stellar occultation)

Their specific requests are the prediction and reconstruction of the occultations.

ELLISMA: (plasma studies)

specific request: prediction of duration of stay in the Ionospheric region. Ephemeris and closest approach to Phobos and Deimos

OMEGA: (infrared spectrometer)

specific request: Nadir pointing coordinates

DYMIO: (ionospheric studies)

specific request: crossing into the Martian magnetosphere

LILAS-2: (Astrophysical experiments)

no specific request

OPTIMISM: (seismology experiments on the small stations)

specific requests: small stations localization and reconstruction of the landing trajectory

METTEG: (meteorology on small stations)

specific request: localization of the small stations

For the Russian side, IKI is responsible of maintaining and updating the NAIF-Space Data base. For example, the ephemeris of the spacecraft will be updated on a weekly basis. Every week, a new file will be cataloged, which will cover a 15 days period: 7 days reconstruction, 7 day extrapolation.

This NAIF Data base will be copied, as soon as any change occurs, in Toulouse, and updated by France with our own results (as lander location). In every french laboratory involved in the mission, the NAIF toolkit will be installed, together with, if needed, special routines developed by our divisions. The Mars94 Naif data base will be cataloged in Toulouse, and updated as frequently as necessary. The scientists will have access at this data base, and will be able to copy, by FTP, the files they want.

8. References:

- [1]: Mars Observer Project- Navigation Plan
JPL Document 642-321 Rev C, June 1990
- [2] Localization process for penetrators and small stations for Mars 94
YA Bojor, V.A. Kotin, Babakine Document, March 1992
- [3] Preliminary studies of a martian localization system
J. Bernard & al., IAF 92-0058

Internal CNES references:

- [3] Mars94: Preliminary results for the balloon localization
J. Bernard, TE/IS/MS/AM/062 (01/1991)
- [4] VAP: Preliminary report on localization
F. Bonneau, TE/IS/MS/AM (08/1992)
- [5] FILON: description of the software
F. Bonneau TE/IS/MS/AM/420 (09/1992)

MERCATOR

N 94 - 23925

METHODS AND REALIZATION FOR CONTROL OF THE ATTITUDE
AND THE ORBIT OF SPACECRAFT

Gilles TAVERNIER, Geneviève CAMPAN
CENTRE NATIONAL D'ETUDES SPATIALES
Toulouse, France

ABSTRACT

Since 1974, CNES has been involved in geostationary satellites positioning. Among different entities participating in operations and their preparation, the Flight Dynamics Center (FDC) is in charge of performing the following tasks :

- orbit determination,
- attitude determination,
- computation, monitoring and calibration of orbit maneuvers,
- computation, monitoring and calibration of attitude maneuvers,
- operational predictions.

In order to fulfill this mission, the FDC receives telemetry from the satellite and localization measurements from ground stations (CNES, NASA, INTELSAT, ...). These data are processed by space dynamics programs integrated in MERCATOR system which is implanted on SUN workstations (UNIX O.S.).

The main features of MERCATOR are redundancy, modularity and flexibility :

- efficient, flexible and user friendly man/machine interface,
- four identical SUN stations totally redundant linked in an Ethernet network.

Each workstation can perform all the tasks from data acquisition to computation results dissemination through a video-network.

A team of four engineers can handle the space mechanics aspects of a complete geostationary positioning from the injection on a transfer orbit to the final maneuvers in the station-keeping window. MERCATOR has been or is to be used for operations related to more than ten geostationary positionings.

Initially developed for geostationary satellites, MERCATOR's methodology was also used for satellite control centers and can be applied to a wide range of satellites and to the future manned missions.

Keywords :

Flight Dynamics, support system, man-machine interface, UNIX O.S., telemetry, maneuvers

1. INTRODUCTION

MERCATOR (Methods and Realization for Control of the ATtitude and the ORbit of spacecraft) is a data processing system used by the Flight Dynamics Center (FDC), which is responsible for space mechanics aspects of geostationary satellite positionings (Launch and Early Operations Phases - LEOP) performed by CNES ground segment located in Toulouse Space Center.

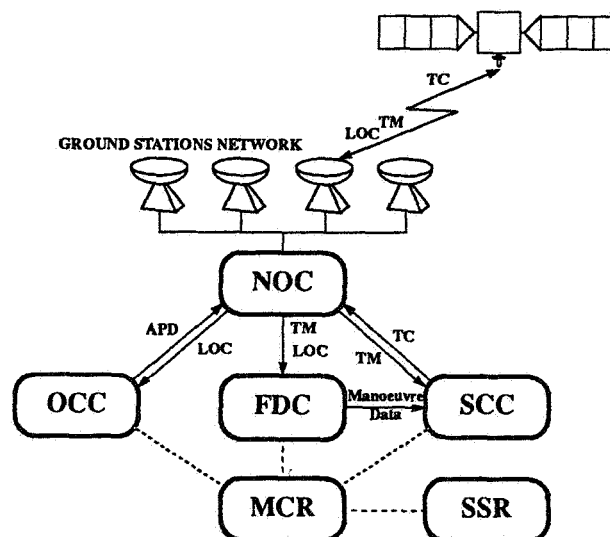


Fig. 1 - CNES ground segment

NOC : Network Operations Center (communications between the different CNES entities involved in the LEOP - video, voice loop, data flows - and with the ground stations all over the world)

SCC : Satellite Control Center (telemetry processing, telecommands)

FDC : Flight Dynamics Center (space mechanics aspects)

OCC : Orbit Computation Center (in charge of stations designations : providing antenna pointing data - APD)

MCR : Main Control Room (mission management)

SSR : Spacecraft Specialists Room

FDC has been involved in LEOP for 18 years :

- 6 positionings between 1974 (Symphonie) and 1988 (TDF 1),

- 10 positionings performed with MERCATOR since 1989 (TELE-X) until more recently, HISPASAT, in September 92.

In charge of the space mechanics aspects, this multimission entity, from the beginning of a project, performs the mission analysis studies, in order to define and verify the feasibility of positioning maneuvers strategies. Then, it develops the space mechanics applications since their conception up to the complete validation of the integrated system. Autonomous entity, it receives the satellite telemetry, and the localization measurements from ground stations via the Network Operation Center (NOC). Its results are transmitted to other entities via video pages. In addition to resolution methods for the various problems encountered during LEOP (space mechanics software), the FDC leans on a data processing system, efficient, flexible and reliable. MERCATOR is usually operated by four space mechanics experts.

FDC is organized on two levels :

- the data processing system named MERCATOR,
- space mechanics applications, some of them multimission, others specific to the mission to achieve.

2. OBJECTIVES OF MERCATOR

MERCATOR was designed in 1986 with some ambitious objectives listed here below (Ref. 1).

First, the data processing system must be simple. It must allow any operational user to have a complete representation of the system and to control it.

Second, concerning the number of operating people, typically three people must be sufficient to perform the flight dynamics tasks related to a complete geostationary positioning. Operational constraints (operations duration, ...) lead to a four-people team in order to insure redundancy.

Another objective is that the same system must apply to a wide variety of spacecraft. Adapting FDC from a previous to a new mission should consist in modification or replacement of elementary modules, either software or ASCII files of the data base (concept of tool box). As several FDC teams might prepare different LEOP simultaneously, FDC facilities have to allow different configurations and an easy and fast transposition from one to another.

The workload corresponding to the implantation of a software must be negligible with respect to its specification, development and validation.

The development must emphasize the assistance to the space dynamics experts, in nominal cases as well as in contingency cases. The interactive monitoring, graphical and diagnosis tools must allow the user to have a complete view of the occurring events through the available data and to extract as much information as possible from them, even if this information is not complete (example : first orbit diagnosis) and leads to rough evaluations.

In contingency cases, the system must allow to insert additional data in reserve software without damaging nominal software.

From the performance and security point of view, it should be possible to filter and condense data before treatment in order to eliminate erroneous data and to limit the treatment computation load.

At the end, and these are constant preoccupations for any project, the development cost must be minimized ... and the probability of success must be ... maximal.

3. ACHIEVEMENT OF THESE OBJECTIVES

3.1 Hardware Choices

Due to their qualities relative to computation power, ergonomics, reliability, cost, ..., UNIX based SUN workstations were chosen.

The modular distributed architecture used for a LEOP is depicted on Fig. 2. It consists of four identical workstations SUN 4/330 (ST1, ST2, ST3 and ST4) linked by a local Ethernet network and two micro-computers (HP Vectra PC1 and PC2) also connected to this network in order to handle the graphical video outputs.

Three workstations are sufficient to perform the operations, the fourth one is only for hot redundancy.

Each workstation hosts the same software and is equipped with :

- central memory of 16 Mbytes,
- bulk memory : one 699 Mbytes disk,
- multiplexer : one ALM2 with 16 channels RS232,
- 150 Mbytes streamer unit,
- extension capability : 2 free slots.

3.2 Software Choices

MERCATOR data processing system can be described as a flight dynamics "support system", i.e. a software structure providing :

- elementary functions (general communication facilities for external data interfaces, data preprocessing, system monitoring, time synchronization, ...),
- shells and state of the art man-machine interface for implantation, modification and implementation of application software.

This structure has been designed so as to minimize the integration effort, most of the work being concentrated on the tool itself.

Moreover, the functions to be performed for different LEOP are similar : consequently, the principle of "a family of flight dynamics specific applications" has been preferred in order to minimize the adaptations of the tools from one mission to another.

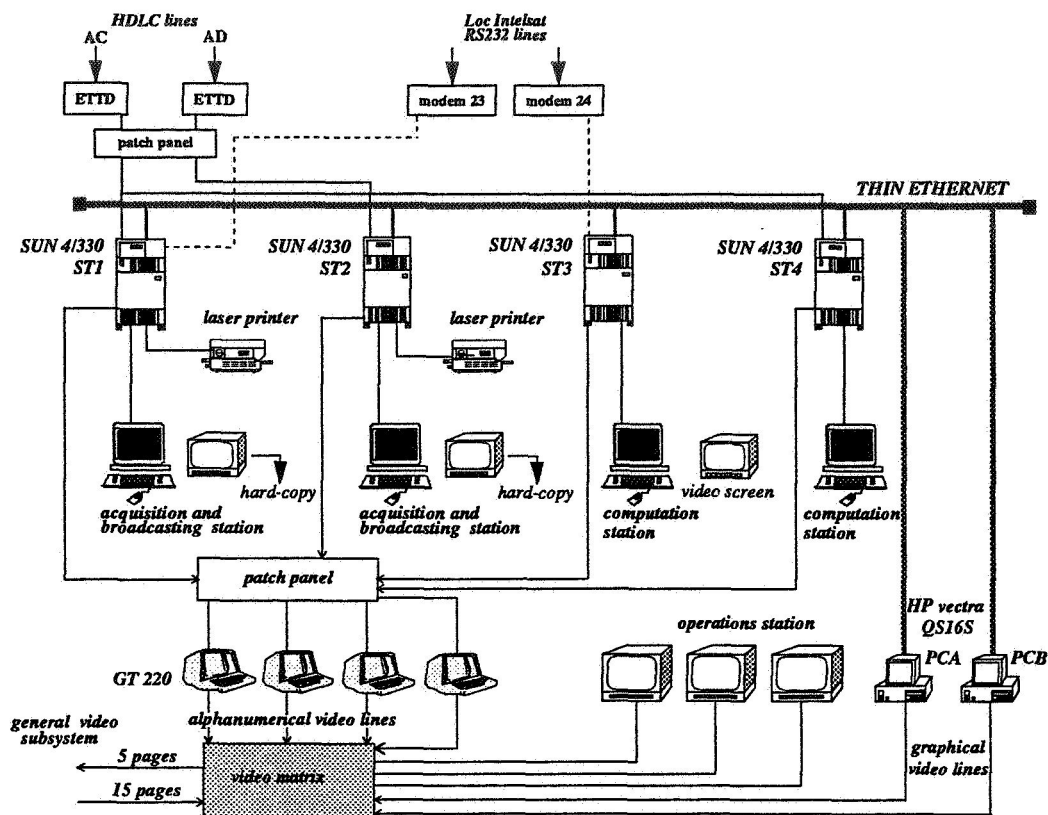


Fig. 2 : FDC Hardware Architecture

3.2.1 Software architecture

Two operating modes are proposed :

. Implantation mode which corresponds to FDC preparation before the LEOP, with two aspects :

- data base management, including classical functions such as creation, modification, suppression, control (consistency or completeness), printing, save, restore, list, ...,

- implantation of applications offering functions such as creation, suppression, modification, control (completeness : control of the presence of necessary files - or information -), consistency with data base (label, units), syntax, list, cross check.

This mode provides complete security and reliability for the implanted applications and data. In addition to this security, great flexibility and simplicity are offered to the user. As an example, the time to implement any new flight dynamics piece of software in Mercator sytem is less than one day.

. Operation mode used during LEOP, consisting in performing the positioning computations and operations in a user friendly and safe environment.

Once the space mechanics applications are implanted in MERCATOR system, FDC can perform the following tasks :

a) MERCATOR applications :

- data acquisition, decommutation and preprocessing (telemetry and localization measurements),
- system monitoring,
- time initialization (simulated time) and synchronization (U. T.),
- various functionalities available to interpret phenomena and results : editions, interactive displays, deferred time executions, chaining of functions, filters on data, visualization and sorting of files, ...),

b) Flight dynamics tasks :

- real-time orbit determination,
- real-time attitude determination,
- computation, real-time monitoring and calibration of orbit and attitude maneuvers,
- operational predictions (eclipses, stations RF visibilities, sensors visibilities, ...).

The performances of the system naturally improve the quality of the man/machine interface.

Moreover, the compatibility of MERCATOR system with UNIX functionalities and multi-windowing allows us to have at our disposal a great number of supplementary analysis possibilities. It has also to be noticed that nowadays, the applications (including mission analysis) are developed on SUN, so there is commonality in terms of work environment since the preliminary studies up to the operations.

The user can open several windows in parallel corresponding to simultaneous execution of different applications.

The software configuration is identical for each workstation. The execution of the different applications is shared out among the workstations set up on the operational network. This sharing is governed by redundancy considerations and by the set of tasks to be done at the given time during the mission.

3.2.2 Main functions of MERCATOR support system

a) Data origin machine

Telemetry and localization data are decommuted and preprocessed after acquisition and stored in both raw and preprocessed forms. This function is usually activated on one or two workstations (for redundancy purposes). The obtained data are available for any application module running on any operational workstation which can define what is its data origin machine. The advantage is that we can specialize the workstations according to the requirements at a given time, but as they are identical on both hardware and software point of view, this configuration can be modified very quickly (a few minutes) when the needs are changing or if there is a workstation failure.

b) System monitoring

The role of this task is to provide operator assistance to the operations. The main functions are :

- . process monitoring :
 - monitoring the running of the different processes activated,
 - monitoring the operating of the various resources (CPU, disks, memory, Ethernet bus, ...),
 - generating alarm detection and messages to inform the operator of any trouble in the progress of processing,
 - archiving all the reports of investigations and monitoring in the Log Book ;

. datation management for the application which allow us to perform simulations or replays in realistic and actual configurations of the positioning operations (simulation of the launch at a certain date, maneuvers, ...), synchronization with the U. T. provided by CNES atomic clock.

c) Data base

Information relative to spacecraft, ground stations and data shared by several applications are organized in a data base structure consisting in different ASCII files, the main ones being :

- . spacecraft technological file :
 - sensors location, orientation, field of view, ...,
 - actuators (apogee motor, thrusters, wheels) location, orientation, characteristics,
 - antenna patterns, frequencies, ...,
- . decommutation table to extract binary values from the telemetry flow :
 - position in telemetry format,
 - occurrences,
 - position of the first bit in the frame, and length,
 - relation with other parameters,
- . telemetry transfer functions to convert binary values into physical values,
- . ground stations file :
 - stations coordinates,
 - meteorological data,
 - equipment delays,
 - calibration data,
- . state file :
 - orbital elements,
 - mass/inertia data,
 - calibration coefficients,
 - ...

These are data computed by some applications and used by others. This file is organized so as to preserve the historical record of the positioning. For instance, when a new orbit is validated in the state file, a new occurrence of the orbital elements is created : parameter label, value, occurrence, date-time of the validation.

d) TM processing

From each telemetry frame of the raw data file, this process decommutes the concerned parameters, then stores them in a file which contains the physical values updated in near real time. The interest of this function quite classical in itself is in the definition of the parameters to be decommuted.

They are defined in two ASCII files (decommutation table and transfer functions - see Data base). This approach has proved its ability to handle various types of satellite telemetry with a minimum time being spent to set the configuration of this process.

e) Input/output data management

There are several ways for the MERCATOR user to define the input data of an application. Each datum is defined by a label, a value and a unit.

When opening the window relative to an application, there is usually a default configuration with :

- default values,
- values taken out from data base files or output files from other applications according to their label.

The user can change this configuration :

- entering a new value,
- changing the name of the file from which the information is extracted (there is an error message "I/O error" if the file does not exist),
- modifying the variable part of the label (the new complete label should exist in the corresponding file, otherwise there is an error message "non initialized value"),
- getting a complete set of input data from a retrieval file created during a previous run of the application : "restore configuration" functionality.

The same mechanism applies to the output data allowing the user to :

- create a retrieval file containing input and output data (label, value) : "save configuration" functionality,
- validate some data in data base or specific files with the same flexibility on the file names and labels.

These various possibilities are very useful during operations and warrant security and consistency in the execution of the different applications, offering at the same time a great flexibility.

f) Chaining of tasks

Applications processing telemetry or localization measurements can be made up of one, two or three modules. These programs can be executed separately or chained (pipe) with the possibility to store the intermediate processed flow into a file (tee).

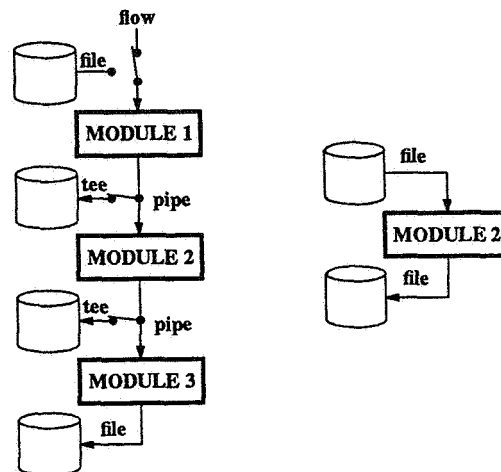


Fig. 3 - Tasks chaining

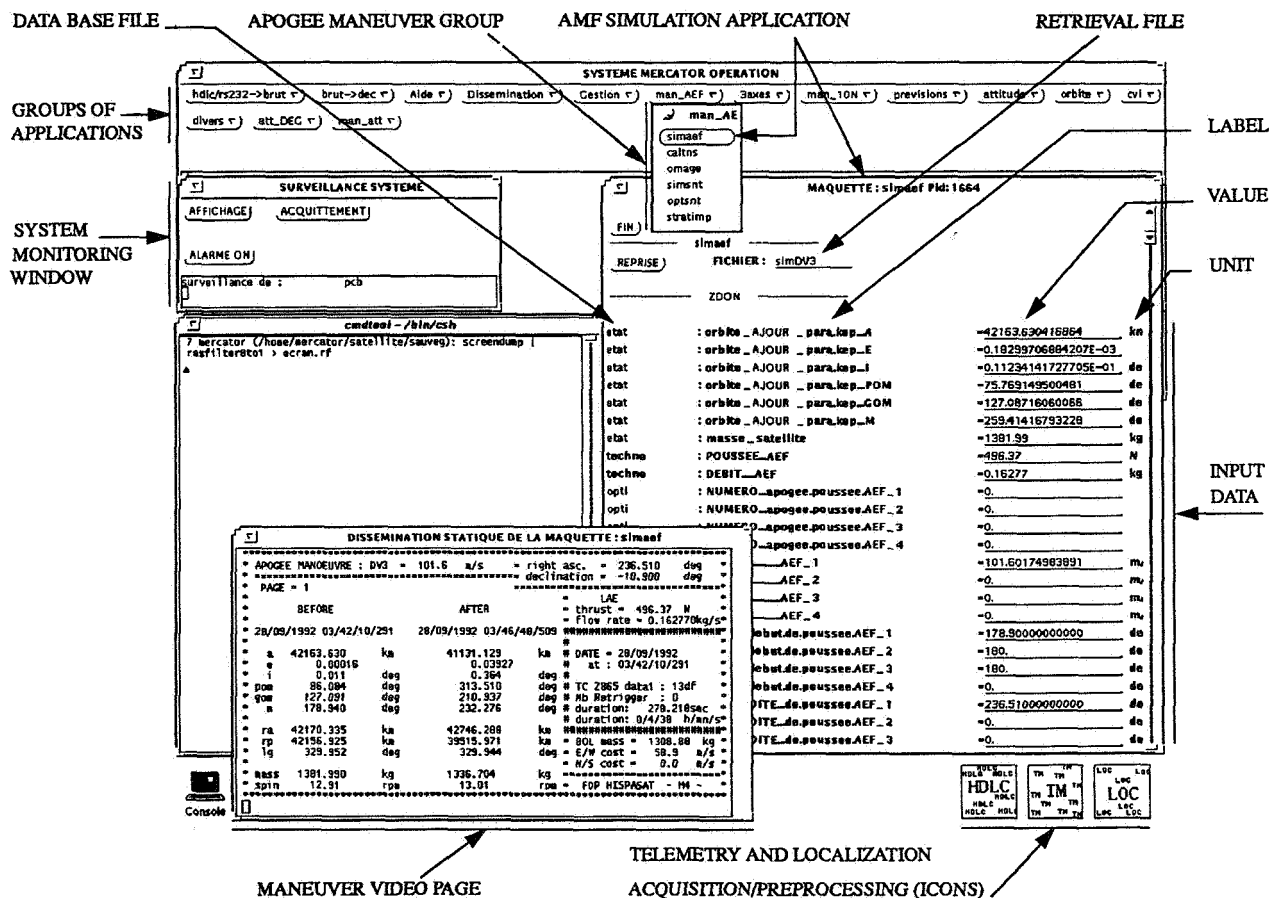


Fig. 4 : Example of MERCATOR screen during operations

The application can be executed in three modes :

- synchronized : real time on flow,
- deferred : off-line replay,
- taking up : starting at given time, then real time on flow when arriving at the end of the file.

The advantage of this approach is its flexibility, allowing to run elementary or complete functionalities in different modes.

g) Video distribution

Results of FDC computations, monitorings, analyses are distributed to other entities via a video network in three forms :

- static alphanumerical pages such as maneuver pages including telecommands to be sent to the spacecraft,

- dynamical alphanumerical pages such as real time display of orbital elements or telemetry parameters,

- graphical displays on telemetry parameters, results of telemetry processing.

These displays are defined in ASCII files prepared in advance but which can easily be modified in real time : it is thus possible to modify the scale of a graphical display without interrupting the application execution.

For more complete information about the results of FDC applications, formatted files can be printed (with possible filter).

CONCLUSION

Designed and developped since 1986, MERCATOR data processing system was validated during TDF 1 LEOP (1988) and operationally operated for the first time in 1989 (TELE-X). Since then, it was successfully used for ten positioning including :

- two LEOP being performed simultaneously (a second hardware configuration was settled for that purpose) : INMARSAT-2 F3/TELECOM 2-A in December 1991 and INMARSAT-2 F4/TELECOM 2-B in April 92,

- a LEOP prepared and successfully performed in a very short time (8 months to be compared to a usual preparation of about 18 months) : ARABSAT 1-C in Februar 92,

- a LEOP performed from an external site (Madrid, Spain) : HISPASAT 1-A in September 92.

Originally designed to perform FDC space mechanics tasks related to geostationary positioning, MERCATOR flexibility and state of the art man-machine interface interested other CNES entities which now use it to perform their own tasks :

- geostationary satellite station keeping : TELECOM 2 and TDF satellite control centers,

- TOPEX/DORIS on-board orbit determination simulation : TOPEX telemetry files processing, DORIS telemetry extraction and treatment (2 Fortran, 3 C and 3 ADA programs, the MERCATOR/ADA interface being developped in 3 days), on two sites : JPL and CNES Toulouse,

- MEPHISTO experiment, part of United States Microgravity Payload (USMP) : telemetry real-time acquisition and decommutation for scientists working in France.

Other entities have developped similar data processing systems or have similar needs :

- low earth orbit satellites ground segments,
- Orbit Computation Center,
- mini satellites ground segments,
- for mission analysis purposes (low earth and geostationary orbits, interplanetary trajectories),
- for manned missions.

So, there are now studies being conducted to develop a new and single man-machine interface and data processing system likely to meet all the present or future requirements related to the space mechanics part of ground segments : OREMUS. At the present time, we have set up a preliminary list of requirements and different solutions based on CNES experience are being investigated.

REFERENCES

1. Belon B., Bergès J.-C., Campan G., Legendre P., "MERCATOR, A new ground system for orbit and attitude control", *Second International Symposium on Spacecraft Flight Dynamics*, Oct. 20-23, 1986
2. Vielcanet P., Moran P., Secretan H., "MERCATOR, A distributed ground control system for geostationary satellite positioning, Ground data system for spacecraft control", *ESA, International Symposium*, June 26-29, 1990
3. Campan G., "Mise à poste INMARSAT-2/F1", *4e Congrès Aérospatial Européen*, Paris, 13-16 mai, 1991
4. Lazard B., "Launch and Early Operations Phases for the dual launch of TELECOM 2-A and INMARSAT-2/F3", *43rd Congress of IAF*, Washington, DC, Aug. 28-Sept. 5, 1992
5. Carrou J.-P., Campan G., Fourcade J., Foliard J., "The technical evolution of ground flight dynamics system utilized for spacecraft operations", *43rd Congress of IAF*, Washington, DC, Aug. 28-Sept. 5, 1992
6. Brousse P., Desprairies A., "HISPASAT launch and early operations phases, Computation and monitoring of geostationary satellite positioning", *Space Ops 92*, Pasadena, California, Nov. 16-20, 1992

Jordan Ellis

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California

N94-23926

Abstract

In the mid 1990s, Russian and Japanese space agencies will each place into highly elliptic earth orbit a radio telescope consisting of a large antenna and radio astronomy receivers. Very Long Baseline Interferometry (VLBI) techniques will be used to obtain high resolution images of radio sources observed by the space and ground based antennas. Stringent navigation accuracy requirements are imposed on the Space VLBI missions by the need to transfer an ultra stable ground reference frequency standard to the spacecraft and by the demands of the VLBI correlation process. Orbit determination for the missions will be the joint responsibility of navigation centers in the U.S., Russia and Japan with orbit estimates based on combining tracking data from NASA, Russian and Japanese sites. This paper describes the operational plans, the inter-agency coordination and data exchange between the navigation centers required for space VLBI navigation.

Key words: Space VLBI, Navigation, Cross-support

1. Introduction

Very Long Baseline Interferometry (VLBI) techniques from ground based antennas have been extensively used to map celestial radio sources. For earth based antennas, the angular resolution is limited by the maximum baseline between radio telescopes which is effectively the diameter of the earth. Longer baselines and thus greatly increased resolution and sensitivity can be attained by placing one of the antennas in earth orbit. In the 1995-96 time frame, Russian and Japanese space agencies each plan to orbit a dedicated radio telescope. The space radio telescopes will observe astronomical radio sources simultaneously with a world wide network of ground radio telescopes. Correlation of the signals received by the space and ground antennas is expected to yield high resolution images of the observed source.

Japan's Institute of Space and Astronautical Science (ISAS) is planning to orbit a 7.2 meter antenna in August '95 using the new M-V booster. The Japanese mission VSOP will be in a 1000 by 20,000 km orbit with a 6.06 hour period and a 31 degrees inclination. In late '95 or early '96, the Russian Astro Science Center (ASC) will orbit a radio telescope with a 10 meter antenna. The Russian spacecraft Radioastron will be placed into a 28 hour orbit with a perigee height of 4000 km, an apogee of 76,800 km and an inclination of 51.5 degrees.

The U.S. will participate in the VLBI science in exchange for tracking and navigation services by the Deep Space

Network (DSN) and co-observation and correlation by facilities of the National Radio Astronomy Observatory (NRAO). Extensive international collaboration between the space agencies and between the multi-national VLBI community will be required to maximize the science return. Schedules for an international network of VLBI radio telescopes will be coordinated to ensure the ground antennas and spacecraft antenna are simultaneously observing the same sources. An international network of ground tracking stations will also support both spacecraft. The tracking stations will record science data downlinked in real time from the orbiting satellites, transfer a stable phase reference and collect two-way doppler for navigation.

Navigation of the Space VLBI (SVLBI) missions is unique in the sense that they are both highly elliptic earth orbiters-with relatively stringent navigation requirements for orbit prediction and orbit reconstruction. Orbit determination will be the joint responsibility of navigation centers in the U.S., Russia and Japan. The primary data type available for navigation is two-way doppler which is generated by the reference phase transfer operation. Since orbit determination for each spacecraft is dependent on using tracking data from several tracking sites, cross-support from the international ground network is essential for meeting accuracy requirements.

This paper describes the operational plans at the U.S. navigation center and the inter-agency coordination and data exchange between the navigation centers required for SVLBI navigation. Elements of this plan include agreements on navigation requirements, tracking strategies, common reference frames and astrodynamics constants; and formulation of compatibility tests to demonstrate the consistency of spacecraft dynamic models, observational models, tracking station locations and orbit estimation techniques. The navigation centers are expected to exchange orbit solutions, tracking data and an assessment of the orbit accuracy. The Mission Control Centers will provide the planned and actual maneuver and antenna pointing data required for modelling the spacecraft dynamics.

2. Navigation Requirements

The Space VLBI missions have science and operational objectives which translate into navigation requirements that are more stringent than those typically encountered for highly elliptic earth orbiters. For the space VLBI missions, there is a need to continuously uplink an ultra-stable ground reference frequency standard to the satellites to provide a phase stable reference for the on-board receiver. A time varying uplink frequency

is transmitted to the spacecraft which is corrected for the predicted doppler shift so that a nearly constant frequency is received at the spacecraft. This signal is coherently retransmitted by the spacecraft to the tracking site where a predicted downlink shift is removed from the received signal to minimize ground receiver tracking loop degradation.

Doppler shifts are computed from the predicted orbit solutions. Accuracy requirements for the predicted orbit arise due to constraints on the maximum round trip phase residual error which can be sampled. The 400 Hz sampling rate used at the NASA tracking stations translates into a maximum downlink frequency error of 175 Hz (3 sigma).

Correlation of the VLBI data received by the orbiting satellites and the ground VLBI network imposes another more stringent level of accuracy requirements on the post flight orbit knowledge. Specific accuracy requirements for the reconstructed orbits are dependent on the characteristics of the VLBI correlator.

Table 1 Navigation Requirements

	Radioastron	VSOP
<i>Predicts</i>		
Position	480 m	170 m
Velocity	46 cm/s	27 cm/s
<i>Knowledge</i>		
Position	90 m	45 m
Velocity	1 cm/s	2 cm/s
Acceleration	$6.8 \times 10^{-8} \text{ m/s}^2$	$3.0 \times 10^{-8} \text{ m/s}^2$

Table 1 summarizes the (one sigma) accuracy requirements which have been by formulated by R. Linfield (Ref 1) assuming specifications for the NRAO/VLBA correlator and the DSN tracking stations. Requirements are expressed as one sigma errors in one dimension. The predicted velocity error represents the component along the tracking station to satellite line of sight and the position error the component along the projection of the velocity vector in the plane of sky. A minimum altitude of 2500 km is assumed for VSOP based on visibility constraints. Reconstruction accuracies are defined for component along the direction to the observed radio source.

3. Tracking System Plans

3.1 Tracking Stations

The VLBI experiment requires lengthy uninterrupted reception of a radio signal at the radio telescope receiving sites. During the VLBI sessions, uplink phase control must be maintained for the ground to spacecraft link. This requirement for continuous contact motivates the use of a world wide tracking network to support science data acquisition, phase transfer and acquisition of radio metric data throughout the orbit.

A combination of Russian, Japanese and NASA tracking stations will provide operations support for the two space VLBI missions. A four station Orbiting VLBI Network is being implemented by NASA. This network includes

a new 11 meter antenna at each of the DSN complexes at Goldstone, Madrid and Canberra; and conversion of an existing 14 meter antenna at the NRAO site at Green Bank. The Russian space agency is upgrading a 32 meter antenna at Ussuriisk to track Radioastron, and ISAS is planning a dedicated antenna at Usuda for tracking VSOP.

The advantages of cross-support from the multi-national network is apparent when considering the station view periods for the space VLBI missions. Radioastron is injected into an orbit with a perigee in the southern hemisphere to maximize coverage from Russian sites.

Since the rate of precession of periapsis is less than 8 degrees/year, (and node less than 15 degrees/year) station view periods for Radioastron are reasonably constant and repeat with a 6 day cycle. During this cycle, the spacecraft is visible only 53 percent of the time from Ussuriisk. The addition of the four NASA stations increases the coverage to 97 percent. Perigee is primarily visible from Canberra.

For VSOP, the situation is very different; perigee is initially in the northern hemisphere to facilitate communication with the booster stage. However the argument of periapsis precesses at a rate of 0.96 degrees/day. Consequently, view periods will change significantly throughout the year. Canberra has the longest view periods after launch and the northern stations 6 months later.

3.2 Tracking Data Characteristics

Each tracking station will be equipped to receive the wideband signal from the spacecraft, transmit a phase reference signal, receive the coherent downlink and generate doppler data for navigation. The Radioastron mission will use a two-way coherent X-band link for phase control and Ku-band for wideband data downlink; the Japanese will use Ku-band for both functions. The NASA stations will accomodate both uplink frequency bands, while the national stations will support only the frequency band for their respective spacecraft.

All stations are expected to be capable of generating two-way coherent doppler for use by navigation. Doppler is routinely collected during the phase transfer process. Consequently nearly continuous coverage is available for the 28 hour Radioastron orbit; and typically 12-22 hours of daily coverage is available for VSOP.

A two-way doppler compensated phase which has predicted uplink and downlink doppler shifts removed is recorded at the receiving stations. Use of this observable for navigation requires recovery of the actual doppler shift and knowledge of the uplink frequency. Since the uplink frequency may change 400 times per second, a two-way doppler shift is constructed based on a pseudo constant uplink. The predicted uplink and downlink doppler shifts are added to the measured phase residual and the resulting doppler shift is normalized to a known constant uplink frequency. NASA stations are expected to generate two-way coherent doppler with a random error of 0.1 mm/sec (1 sigma) for a 60 second count time

and a systematic bias of 0.1mm/sec. The latter is due to the phase reconstruction procedure.

In addition to the two-way doppler generated during the phase transfer process, other data types may be available for determining the orbit. A C-band radio link is used for spacecraft operations by the Radioastron Mission Control Center which is located in Eupatoria. During the daily commanding sessions, three 8-10 minute passes of two-way C-band range and doppler are typically scheduled. ISAS will command and receive spacecraft engineering data from VSOP using an S-band link from Kagoshima. Nearly continuous two-way range and doppler will be received during these passes. VSOP will also carry a GPS receiver which can directly provide estimates of the orbit during GPS visibility periods. However, the GPS receiver is considered primarily a technology demonstration.

4. Navigation Covariance Study Results

Independent navigation studies were conducted by the three agencies to determine tracking strategies for satisfying accuracy requirements and identifying dynamic and observational modelling errors limiting the navigation performance (Refs 2-5). Orbit determination accuracies were evaluated for different data strategies, assuming unmodelled errors due to solar pressure, attitude control accelerations, earth's geopotential, station locations, doppler bias and media effects. The studies confirmed the feasibility of satisfying requirements using primarily the extensive two-way doppler collected throughout each orbit as part of the phase transfer process.

A brief summary of the principal conclusions are as follows:

1. For Radioastron, navigation accuracy requirements are satisfied with a tracking span that includes at least 22 hours of X-band doppler data for one 28 hour orbit collected from at least two sites. Data from only a single site is not adequate. Orbit accuracies are considerably enhanced if tracking data can be collected in the vicinity of periapsis (ie $\pm 3hrs$) where the dynamic changes are a maximum. This criteria can be met with C-band ranging passes during commanding sessions or doppler data from Canberra.
2. The VSOP requirements are met by processing doppler data collected for a 24 hour span covering four orbits. Typically, the data span may include up to 4.5 hours per orbit or 18 hours for a 24 hour span. It should be emphasized that the requirements need only be satisfied for segments of the orbit where it is feasible to collect VLBI data. This is fortunate since predicts requirements are more stringent at lower altitudes and would be difficult to satisfy below 2500 km.
3. The dominant unmodeled errors are due to solar pressure acceleration errors, and errors introduced by station locations and the doppler bias. Plans are being formulated by the navigation agencies to reduce the effect of these error sources. Comprehensive solar pressure models have been developed and an effort is

underway to measure the reflectivity properties. An operational interface has been established to utilize the science viewing schedule for predicting the orientation of the spacecraft antenna which in turn influences the solar pressure effects. Efforts are also underway to use GPS and VLBI techniques to determine the location of the tracking stations in a consistent reference frame. Improved station location estimates of Ussuriisk and Usuda will be determined from VLBI observations between antennas at these sites and at DSN sites

5. Navigation Support Plans

5.1 Agency Roles

Orbit determination will be the joint responsibility of navigation centers in the U.S., Russia and Japan. The Flight Control Center in Kalliningrad and JPL's Multimission Navigation Center in Pasadena will share responsibility for determining the Radioastron orbits using data from the Russian tracking sites and from the four NASA sites. Similarly, the JPL navigation center and the ISAS navigation center in Sagami-hira will be responsible for orbit determination for VSOP using data from the Japanese and NASA sites. GPS based solutions may also be available for VSOP.

Orbit solutions, independently determined by each center, will be exchanged, solution differences resolved and the accuracy of the solutions evaluated. The overall objective of this exchange is demonstrate the consistency of the solutions computed by each agency. It is expected that each navigation center will be responsible for providing predicted orbits for use at their respective tracking sites and reconstructed orbits for use at their national VLBI correlation center.

5.2 JPL Operations Plans

JPL navigation will independently determine Radioastron and VSOP orbits using tracking data from all supporting sites. Predicted orbits for a seven day period will be computed twice per week and distributed to the DSN tracking sites for acquisition and phase transfer. Orbit files will be delivered using the standard DSN interface. State vectors will also be provided to the Russian and Japanese navigation centers.

Post flight (reconstructed) orbits will be computed bi-weekly using all available tracking data. The resulting ephemerides will be delivered to the U.S. Space VLBI project center for distribution to the NRAO VLBI correlator and to other potential users. Ephemerides will be distributed in a standard portable format using JPL's Navigation Ancillary Information Facility to prepare the files. Figure 1 illustrates the navigation data flow for the JPL center.

Reconstructed orbit solutions for Radioastron will cover one 28 hour orbit period and will be determined by fitting data collected during a 28-32 hour span. VSOP orbits will be based on fitting data over four orbits. The correlation activity is also expected to provide the navigation centers with an independent assessment of the orbit accuracies.

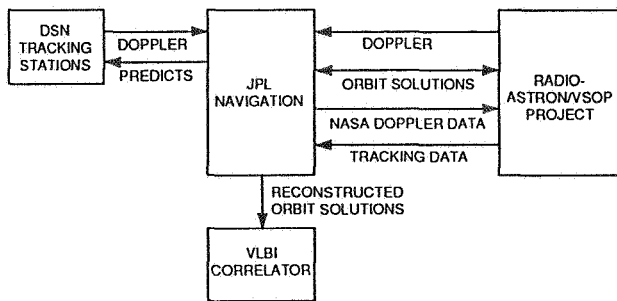


Fig 1. DSN Navigation Data Flow

5.3 Data Exchange Plans

This multiagency orbit determination requires extensive coordination between the navigation centers and the Mission Control Centers. Long term planning information on spacecraft events that affect the spacecraft dynamics will be provided by the Mission Control Centers. This includes maneuver plans, a schedule of radio sources to be viewed, and spacecraft antenna and solar panel orientation data. A 30 day schedule will be transmitted twice per month for use in the computation of predicted orbits.

The navigation centers will exchange state vectors and all tracking data collected by their respective sites twice per week. Tracking data will be corrected for troposphere and ionosphere effects before transmission. Updates of actual antenna pointing and maneuver data will be also be distributed at this time. JPL navigation deliverables will be stored in a data base maintained by the U.S. Space VLBI Project which will be accessible by external users.

6. Compatibility Test Plans

A key task in this cooperative navigation effort was to define a standard set of constants, dynamic models and reference frames to be used by the navigation centers. At the outset, the agencies agreed to adopt the J2000 reference frame, a DE200 planetary ephemeris and a GEM T2 geopotential model. Astrodynamics constants, time transformations, solid earth and ocean tidal models are to be based on the standards in IERS Technical Note (1989).

A series of studies were initiated to identify dynamic effects which must be modeled to satisfy accuracy requirements and to demonstrate the compatibility of the trajectory propagation procedures. It was agreed to include Newtonian point mass accelerations for planets and moon, relativistic perturbation accelerations for the Sun and Earth, a geopotential of at least order 17 and solid earth and ocean tides and solar radiation pressure. Atmospheric drag effects will also be modeled for VSOP.

Procedures for modeling the solar radiation pressure accelerations due to the spacecraft antenna, solar panels and bus have been adopted and results of the software implementation are currently being compared. The Lavochkin Association is tasked to determine the reflectivity properties of the components of the Radioastron spacecraft. ISAS will provide similar data for VSOP.

A three stage compatibility test program is being pursued to demonstrate that consistent orbit determination results can be obtained by the three navigation teams. This plan is similar to the one followed by the U.S. and Russian navigation centers in preparation for supporting the Mars PHOBOS mission. The first phase entails detailed testing of the dynamic models assumptions by comparing results of trajectory propagation. The second phase will be to validate the observational modelling assumptions used for processing the exchanged radio metric data. Detailed debugging type comparisons will be made of all the intermediate quantities generated in the computation of a single computed observable. Data for a representative earth orbiter will also be processed by each facility and data residuals compared. The final test is an end-to-end test designed to demonstrate the consistency of the estimation procedures and to validate the data exchange formats and procedures.

Concluding Remarks

With expected mission lifetimes of 3-5 years, the space VLBI missions represent one of the most intensive and long term applications of the concept of cross-support. It is only with the support from an international ground network and the collaborative efforts of the multi-agency navigation centers that full benefits of the mission science may be realized.

References

1. Linfield, R. "Orbit Determination Requirements for VSOP and Radioastron" JPL Internal Memo, June 6, 1991.
2. Ellis, J., and Estefan, J.A. "An Assessment of Soviet Capabilities for Radioastron Navigation", JPL IOM 314.7-119, Feb.25, 1991.
3. Pavlov, V et al, "Spectr-R Mission Control Organization", S.A.Lavochkin Association, ASC, JPL/BC, March 1992.
4. Estefan, J., Christensen, C. and Ellis, J., "Japanese and DSN Orbit Prediction Capability for the VSOP Mission", JPL IOM 314.5-1551, July 1, 1991.
5. Ichikawa, T., "Japanese VSOP Navigation and Orbit Determination", ISAS presentation at VSOP Working Group Meeting, Tokyo, June 1992,

N94-23927

THE GEO-CONTROL SYSTEM FOR STATION KEEPING AND COLOCATION OF GEOSTATIONARY SATELLITES

O. Montenbruck¹, M. C. Eckstein¹, J. Gonner²

¹German Space Operations Center, DLR, DW-8031 Oberpfaffenhofen, Germany

²Société Européenne des Satellites, L-6832 Betzdorf, Luxembourg

ABSTRACT

GeoControl is a compact but powerful and accurate software system for station keeping of single and colocated satellites, which has been developed at the German Space Operations Center. It includes four core modules for orbit determination (including maneuver estimation), maneuver planning, monitoring of proximities between colocated satellites and interference and event prediction. A simple database containing state vector and maneuver information at selected epochs is maintained as a central interface between the modules. A menu driven shell utilizing form screens for data input serves as the central user interface. The software is written in Ada and Fortran and may be used on VAX workstations or mainframes under the VMS operating system.

Key Words: Station keeping, colocation, geostationary satellites, software.

1. INTRODUCTION

Control centers of geostationary satellites are facing a situation today which is considerably different from that experienced one decade ago. Most satellites have to be controlled within narrow boxes of $\pm 0.1^\circ$ width and many owners even intend to reduce the control box to $\pm 0.05^\circ$ to keep a satellite within the field of view of non-steerable Ku-band antennae. At the same time, the continued effort to improve and extend the international satellite telecommunication network has almost led to an overpopulation of certain parts of the geostationary ring. In Europe, for example, the World Administrative Radio Conference (WARC) has assigned common longitude slots for up to eight satellites from different nations. Furthermore, many satellite owners intend to offer extended direct broadcasting capabilities, which can presently only be

met by placing multiple satellites into a common window. Colocation, i.e. station keeping of multiple satellites in common or adjacent control boxes, has therefore become a major requirement for satellite operations near the turn of the century. The need for an increased station keeping performance is further complemented by the requirement to reduce cost driving factors such as operator work load or software maintenance efforts.

Since presently most of these needs are not met by available software packages, the German Space Operations Center (GSOC) has developed a new system to facilitate and improve station keeping and colocation operations of geostationary satellites (Ref. 1). It is based on GSOC's long-term experience in mission control and operations and meets design criteria like

- reduction of operator effort by a high degree of automation,
- easy usage for operators, spacecraft engineers and analysts,
- visualization of results by elaborate graphical output,
- multi-satellite operations concept,
- high-precision modeling,
- simple overall architecture and
- high software quality standards (Ref. 2).

GeoControl consists of the core modules (ORBIT, MAPLA, EVENT and COLO), which are completely satellite and station independent (Fig. 1). The four modules support the basic tasks of

- orbit prediction and determination including maneuvers
- maneuver planning for regular station keeping, station acquisition, shifts in longitude, proximity avoidance or reorbiting
- ephemeris and event prediction including long-term forecasts of shadow transits and interferences

2. FEATURES

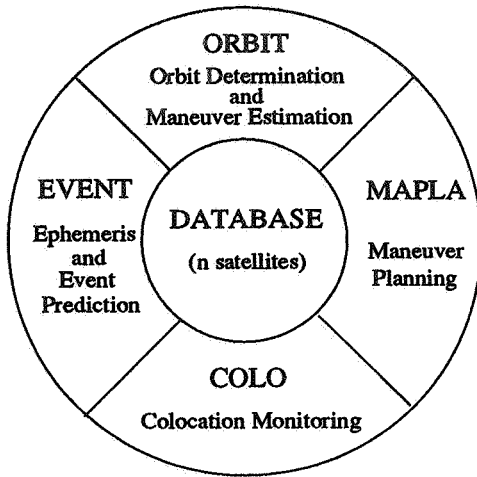


Fig. 1. GeoControl architecture

- relative motion monitoring and proximity analysis

All modules are completely independent of each other and exchange orbit information only via a common state vector and maneuver database. The well-structured architecture of *GeoControl* provides for its easy extension by station and satellite specific software components that may e.g. be required for tracking data preprocessing, antenna steering, maneuver realization or orbit telex generation.

GeoControl is presently employed to control the following satellites:

- TVSat-2 ($-19.2^\circ \pm 0.10^\circ$); loosely collocated with Olympus-1 ($-19.0^\circ \pm 0.10^\circ$) and TDF-1/TDF-2 ($-18.8^\circ \pm 0.10^\circ$);
- Astra-1A and -1B collocated at ($+19.2^\circ \pm 0.10^\circ$); to be collocated with Astra-1C, -1D and -1E in 1993-95;
- DFS-1 ($+23.5^\circ \pm 0.07^\circ$) and DFS-2 ($+28.5^\circ \pm 0.07^\circ$); supplemented by DFS-3 ($+33.5^\circ \pm 0.07^\circ$) in 10/92.

GeoControl has further been used to support the launch and early orbit phase and the station acquisition of Eutelsat II-F1, -F2, -F3 and -F4 (see Ref. 3) and DFS-3.

2.1 Orbit Determination

The ORBIT program provides orbit determination and impulsive Δv estimation capabilities for transfer orbit, drift orbit and on-station operations. Supported tracking data types include Azimuth/Elevation and X/Y-angles in addition to range and two-way Doppler measurements. Precision orbit and measurement modeling ensures an accuracy that is only limited by tracking capabilities and on-board attitude thruster activities (Fig. 2).

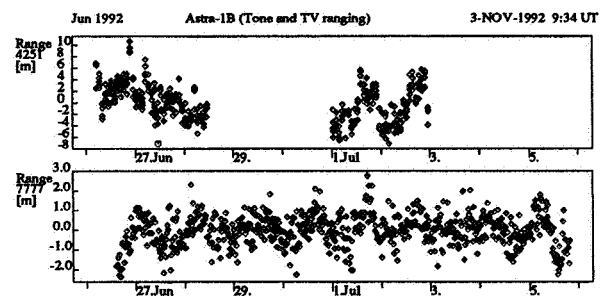


Fig. 2. Sample residuals plot illustrating meter-level orbit determination accuracy obtained over a 9 days tracking arc for Astra-1B. (Top: Tone-ranging system. Bottom: TV-ranging system.)

Individual maneuver components can be estimated during an orbit determination along with orbital elements, radiation pressure coefficients and measurement bias values. This allows for a preliminary maneuver calibration with as little as 3-6 hours of tracking data after a maneuver and enables a fast response in case of maneuver execution errors.

Important results are presented in comprehensive graphs (residuals plots (Fig. 2), station keeping plots (Fig. 3), station acquisition plots) and an orbit determination summary sheet to keep text output to a minimum. These features not only result in an acceleration of operational procedures but also in a considerable reduction of the amount of data that has to be archived for off-line mission analyses.

2.2 Maneuver Planning

The MAPLA program plans the velocity increments and times of impulsive maneuvers for a full station keeping cycle in a single run. Up to three

East/West and two North/South maneuvers per cycle are computed analytically from the difference of predicted orbital elements and target elements at the cycle end. Target elements may either be based on a built-in station keeping strategy or provided manually by the user to support all types of regular and non-regular maneuvers (e.g. shifts in longitude). Maneuver time constraints and deterministic cross-couplings are considered in the maneuver planning process to meet the needs of precision station keeping and colocation with realistic thruster systems (Ref. 4).

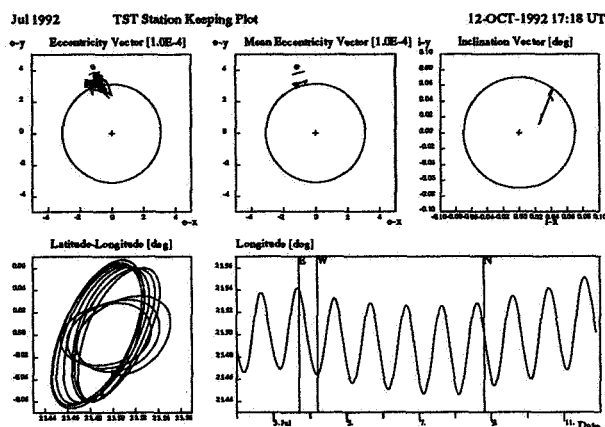


Fig. 3. Sample station keeping plot

Maneuver planning results are summarized in a station keeping plot (Fig. 3) for immediate valuation by the operator. It illustrates the satellite's motion with respect to the longitude and latitude deadband together with the evolution of the eccentricity and inclination vectors. The station keeping plot therefore provides a convenient visualization of all quantities that are required to assess the evolution of the satellite orbit and the impact of planned station keeping maneuvers.

2.3 Ephemeris and Event Prediction

Ephemeris and event predictions required during on-station operations may conveniently be generated by the EVENT program. Supported products include

- Cartesian and Keplerian ephemerides,
- antenna pointing predictions,
- Sun-spacecraft-Earth and Moon-s/c-Earth angle predictions,
- true spacecraft local time computation,

- window violation warnings,
- predictions of Earth or Moon shadow transits,
- predictions of Sun or Moon interferences with infrared Earth sensors (Fig. 4) ,
- predictions of Sun interferences with the ground station antenna beam.

The generation of station keeping plots (Fig. 3) over a sequence of cycles is, furthermore, supported for long-term maneuver planning and easy report generation.

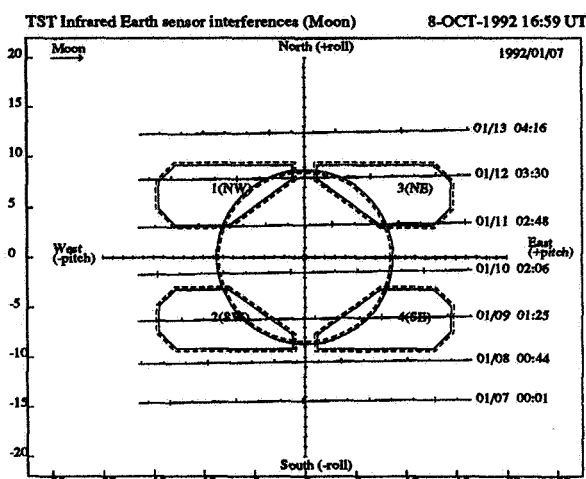


Fig. 4. Sample sensor interference plot

Aside from basing a prediction on the best-known orbit and maneuver database the user may choose a center-of-box ephemeris for long-term forecasts. Approximate shadow transits or Sun/Moon interferences can thus be predicted long in advance before the actual or planned satellite orbit is known.

2.4 Colocation

As mentioned in the introduction, colocation operations are supported as an integrated feature of the *GeoControl* system. This is achieved by the design of the MAPLA maneuver program and by the dedicated COLO program.

By means of appropriate target element biases, MAPLA may be used to implement different types of separation strategies (e.g. longitude or eccentricity/inclination separation) without losing the freedom to plan each satellite's maneuvers independently. Similarly MAPLA may be used to compute evasive maneuvers that may e.g. be required after maneuver failures or during close approaches

with uncontrolled satellites in the event of a longitude shift. Colocation operations can thus be performed at a modest increase of operator workload compared to station keeping of individual satellites.

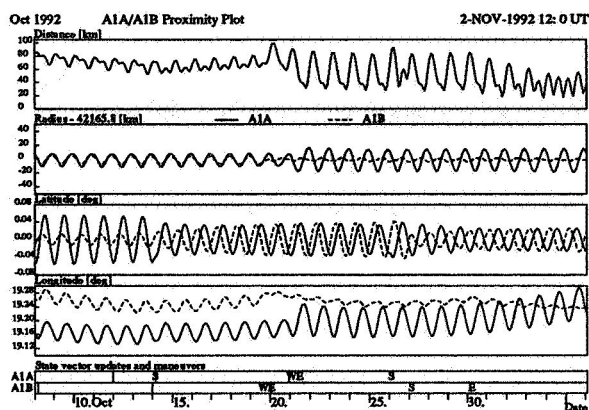


Fig. 5. Sample relative motion plot for colocated satellites

The MAPLA maneuver planning program is supplemented by the COLO relative motion monitoring program. It supports the detection and valuation of proximities in case of longitude swap or colocated satellites. Both text and graphics information is provided for detailed analyses and avoidance maneuver planning. A sample relative motion plot representing the switch of ASTRA-1A and 1B from the longitude to the eccentricity and inclination separation colocation strategy is shown in Fig. 5.

3. THE GEO-CONTROL DATABASE

The open architecture of the *GeoControl* system is based on the concept of an orbit and maneuver database, which serves as the only interface between different programs. All core modules have internal capabilities for generating a satellite's trajectory from state vector and maneuver information. Thus *GeoControl* does not require any interface ephemeris files. The database itself is a fixed format ASCII text file that can easily be read and modified by a standard text editor and no commercial database software (dBase, Oracle) is therefore required to maintain the database.

Two different functional types of database files are maintained within *GeoControl* to meet the needs of practical station keeping operations. A *scratch* database is allocated for each satellite, which collects database records generated by the ORBIT

and MAPLA programs in the course of an orbit determination or maneuver planning. The records created by these programs are appended to the scratch database, which thus contains an incremental log of all results obtained by the operator during a station keeping cycle. In addition to the scratch database, the *GeoControl* system supports a *best-knowledge* database, which provides a time ordered sequence of database records that represent the best orbit information available for contiguous time intervals. The maintenance of the best-knowledge database is supported by an interactive program that allows the user to select a database record from the scratch database and insert it into the best-knowledge database. The program automatically inserts the record at the appropriate position and replaces existing records covering the same time slot. Thus, the best-knowledge database contains the full history of the satellite orbit in a compact format and allows for a simple generation of ephemerides, event lists, station keeping plots and relative motion plots over arbitrary periods of time when using the EVENT and COLO programs.

4. USER INTERFACE

The core programs of the *GeoControl* system are operated via command files and can be executed as batch or foreground jobs. The operator's interaction with *GeoControl* is essentially limited to the menu driven configuration of the job command files using a fully interactive, forms-oriented setup creation software.

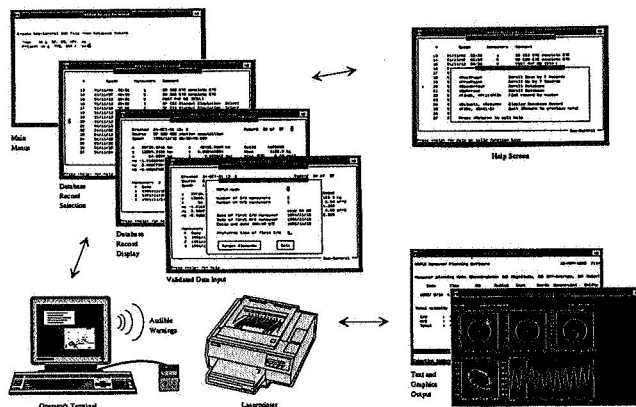


Fig. 6. Interactive *GeoControl* user interface

This software allows the user to select an epoch state vector from the orbit database and enter

those parameters that have to be changed from one run of the software to the next (e.g. the tracking interval for an orbit determination). Input parameters that have to be modified at rare instances only are stored in template setup files. All menus and input forms are implemented using DECforms, which provides for easy usage and elaborate input validation.

5. SYSTEM REQUIREMENTS

GeoControl may be installed on arbitrary VAX workstations or multi-user systems running the VMS operating system with a minimum of 4 MB core memory and 20 000 blocks of free disk space. The system may be operated from workstation consoles and alpha terminals. DEC Ada, DEC Fortran and DECforms are required to rebuild the software from the source code, which is delivered with the software. A run-time license for DECforms is required on each machine running *GeoControl*. All graphics output is generated in PostScript language by a built-in graphics package and can be printed on standard laser printers or displayed on workstation terminals. Other graphics packages like the GKS Graphical Kernel System can be supported as an option.

6. OPERATIONAL EXPERIENCE AT SES

The present section summarizes the experience of the Société Européenne des Satellites (SES) in using *GeoControl* for the station-keeping of the ASTRA satellites. The ASTRA spacecraft are three-axis stabilized direct broadcasting TV satellites colocated at the orbital position of $+19.2^\circ \pm 0.10^\circ$ East longitude. ASTRA-1A (a GE Series 4000) was launched in December 1988 and ASTRA-1B (a GE Series 5000) was launched in March 1991. The future plans of SES are to extend the present configuration to four colocated satellites. The next two satellites, ASTRA-1C, 1D and 1E, will be Hughes HS-601 spacecraft to be launched in 1993, 1994 and 1995, respectively.

After a trial phase of three months that started in January 1992 and during which efficient training and support have been provided by GSOC, *GeoControl* has become the primary station-keeping software used by SES. During this trial phase, *GeoControl* has been run in parallel with other station-keeping softwares and has proved superior

to those in terms of the various design criteria listed in the introduction.

In particular the open architecture of the software and database concept described in section 3 have been greatly appreciated. The user interface has proved effective and the extremely reduced number of steps required to generate any kind of output results (plots, summary sheets, detailed outputs) confirmed superiority of *GeoControl* in terms of operations workload.

As already mentioned in section 2.1, orbit determinations performed based on tracking data from a high precision TV-ranging device demonstrated the accuracy of the propagator used by the ORBIT program. In the context of colocation another key advantage of ORBIT is the ability to evaluate the performance of maneuvers shortly after their execution, which increases operations safety. Finally, a remarkable feature of the orbit determination module is that it allows to solve for *two* components of the solar radiation force (along and normal to the Sun-spacecraft line). This greatly improves orbit determination accuracies in the case of ASTRA-1A, whose solar panels are not directly facing the Sun.

The capability of MAPLA to take deterministic cross-coupling into account in maneuver computations also proved to be an important advantage for the colocation of three-axis stabilized spacecraft. In the particular case of the ASTRA satellites, which are of different designs, the independence of the MAPLA module from the peculiarities of thruster systems is essential. By contrast to most orbital softwares *GeoControl* also offers a sufficiently general maneuver planning concept to support different station keeping strategies. As an illustration, MAPLA allows eccentricity and longitude control of the ASTRA satellites by alternating station-keeping cycles using single and double East/West corrections. The flexibility of MAPLA also clearly appeared when the colocation strategy used to colocate ASTRA-1A and 1B was changed from longitude separation to eccentricity and inclination separation in October 1992. Proper setting of parameters permitted MAPLA to compute the maneuvers required to proceed with the configuration switch as well as regular station-keeping maneuvers in the case of both strategies.

The transition from one strategy to another is illustrated by the relative motion plot of Fig. 5. In addition to such plots the relative motion monitoring program COLO produces text output con-

taining an analysis of the intersatellite separation in terms of the error ellipsoid concept. This allows actual tracking, orbit determination and expected maneuver errors to be taken into account in the evaluation of the separation safety.

To summarize, usage of *GeoControl* at SES confirmed that the software has been designed on the basis of a long-term experience in station-keeping of geostationary satellites and includes state-of-the-art station-keeping and colocation techniques. Finally, the application of high software quality standards, resulting in simple maintainability and installation of *GeoControl* and highly understandable code, has been greatly appreciated by SES.

ACKNOWLEDGEMENTS

We express our special thanks to M. Esser, A. Härting, J. Hermann, F. Karch, H. Meixner and S. Jones for valuable contributions during the development, implementation and testing of *GeoControl*. Their long-term experience in station keeping of geostationary satellites has been a driving factor for the design of an easy to use system that meets the needs of practical operations. We are further indebted to P. Francken, H. Laroche and P. Wauthier for a number of suggestions that helped improving the system.

REFERENCES

1. Pietraß A., Eckstein M. C., Montenbruck O., 1991. *Software for Station Keeping of Co-located Geostationary Satellites*, Proceedings IVth European Aerospace Conference (EAC 91), ESA SP-342, pp. 329-335.
2. *GSOC Software Engineering Standards: Vol. 3: GeoControl Core and Extension Modules*, DLR-GSOC.
3. Montenbruck O., Gill E., 1991. *Estimation of Attitude Thruster Activity of Spacecraft in Sun Acquisition Mode*, Proceedings of the ESA Symposium on Spacecraft Flight Dynamics, ESA SP-326, pp. 275-278.
4. Eckstein M. C., 1990. *Geostationary satellite orbit control considering deterministic cross-coupling effects*, 4th IAF Congress, Paper IAF 90-326, Dresden.

N94-23928

HISPASAT LAUNCH AND EARLY OPERATIONS PHASES COMPUTATION AND MONITORING OF GEOSTATIONARY SATELLITE POSITIONING

Pascal BROUSSE
CENTRE NATIONAL D'ETUDES SPATIALES
Toulouse, France

Arnaud DESPRAIRIES
MATRA MARCONI SPACE
Toulouse, France

ABSTRACT

Since 1974, CNES, the French National Space Agency, has been involved in the geostationary Launch and Early Operations Phases (LEOP) of moving satellites from a transfer orbit delivered by a launcher to a geostationary point (Symphonie A and B, Télécom 1-A, -B and -C, TDF 1 and 2, TELE-X, INMARSAT-2/F1, F2, F3 and F4, ARABSAT 1-C, Télécom 2-A and -B, HISPASAT 1-A).

During the operations and their preparation, the Flight Dynamics Center (FDC), part of CNES LEOP facilities, is in charge of the space mechanics aspects.

What is noteworthy about the Spanish HISPASAT satellite positioning is that all the operations were performed on the customer's premises, and consequently the FDC was duplicated in Madrid, Spain.

The first part of this paper is the FDC presentation : its role, its hardware configuration, and its space dynamics ground control system called MERCATOR. The second part of this paper details the preparation used by the FDC for the HISPASAT mission : hardware and software installation in Madrid, integration with the other entities and technical and operational qualifications. The third part gives results concerning flight dynamics aspects and operational activities.

Key words : geostationary satellite positioning, space mechanics, operations, ground control system, man machine interface.

1. THE FLIGHT DYNAMICS CENTER

1.1 Entities Involved in the LEOP

Different entities of the CNES participating in operations are described in Fig. 1.

For all the LEOP missions before HISPASAT 1-A, these entities including the SCC were on the CNES premises in Toulouse, France. At the end of the LEOP, the satellite was delivered to the customer's Satellite Control Center.

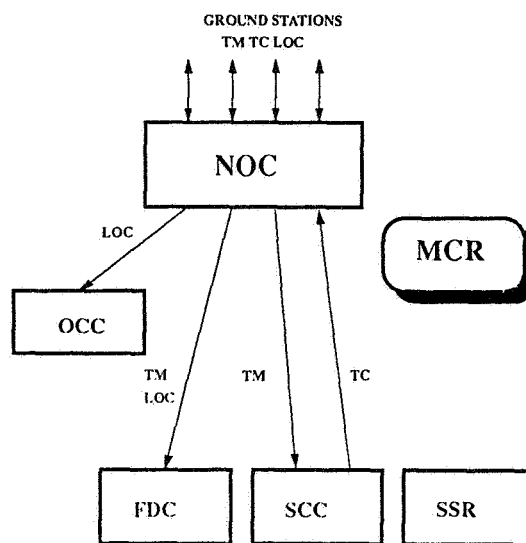


Fig. 1

NOC : Network Operations Center (communications with the ground stations all over the world)

SCC : Satellite Control Center

FDC : Flight Dynamics Center (space mechanics aspects)

OCC : Orbit Computation Center (in charge of station designations : providing satellite azimuth and elevation to the ground stations)

MCR : Main Control Room (mission management)

SSR : Satellite Specialists Room

TM : satellite telemetry

TC : telecommand

LOC : localization measurements

1.2 FDC Role and Organization

Among the above entities, the FDC team is responsible for the development of the methods necessary for the resolution of the space mechanics problems during the LEOP.

First, at the beginning of a project, the FDC team issues the flight dynamics mission analysis to prove the feasibility of the positioning in 99,73 % (3 sigma) of the cases of dispersions while fulfilling the satellite and mission constraints and minimizing the propellant mass consumed.

This document gives the results of computation concerning :

- the launch window meaning the authorized time of launch every day of the year,
- apogee maneuvers strategy with a back-up strategy for each maneuver,
- prevision for events (solar eclipses by the earth and moon, sensors blindings, visibilities),
- orbit determination accuracy,
- attitude determination accuracy,
- station-keeping maneuvers strategy.

The second task, before operations, is the development and the validation of the space mechanics software.

During the LEOP, the FDC performs the following tasks :

- orbit determination,
- attitude determination,
- optimization, computation, monitoring and calibration of attitude maneuvers,
- optimization, computation, monitoring and calibration of orbit maneuvers,
- real-time orbit determination with Kalman filter during apogee maneuvers if Doppler measurements are performed,
- operational predictions as geometric and radio-frequency contact, sensors visibilities, solar eclipses by the earth or moon.

Real-time telemetry inputs are received from the satellite while localization measurements are received from the ground stations. Other inputs include technological and mission data given by the satellite manufacturer.

Results of computation include maneuver parameters, graphic monitorings and operational predictions which are transmitted to the other entities involved in the operations through video pages.

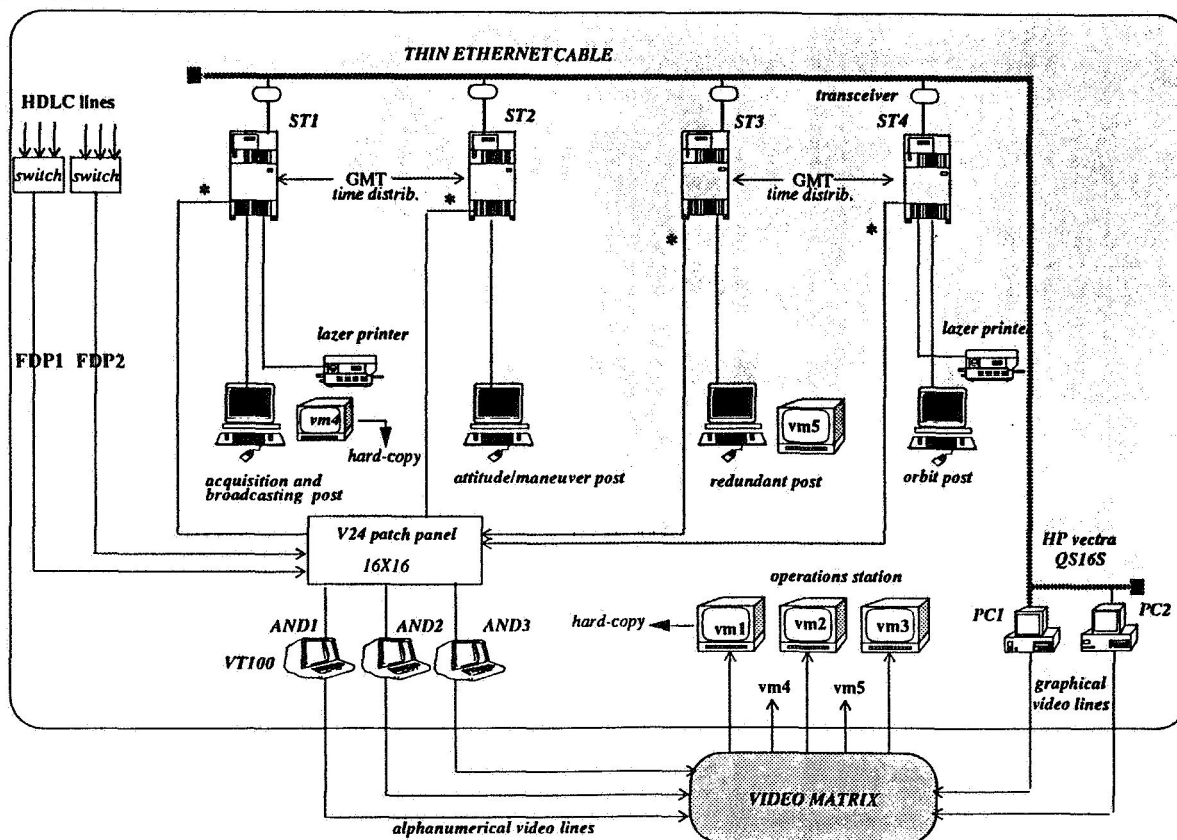
1.3 Hardware Configuration

The hardware configuration depicted in Fig. 2 consists of four identical workstations SUN 4/330 (ST1, ST2, ST3 and ST4) linked by a local Ethernet network and two micro-computers (HP Vectra PC1 and PC2) connected to the network in order to handle the graphic video outputs.

Three workstations are sufficient to perform the operations, the fourth one is only for hot redundancy.

Each workstation hosts the same software and is equipped with :

- central memory of 16 Mbytes,
- bulk memory : one 669 Mbytes disk,
- multiplexer : one ALM2 with 16 channels RS232,
- 150 Mbytes streamer unit,
- extension capability : 2 free slots.



vm: video monitor
AND:alphanumeric display

* 4 lines: 3 from ALM2 and 1 from serial port A

Fig. 2

Each workstation has the following peripherals :

- alphanumeric console VT220 for generation of video display,
- laser printer,
- Universal Time equipment.

Each micro-computer HP Vectra is equipped with :

- central memory of 1 Mbyte,
- floppy disk unit of 1.2 Mbyte,
- hard disk unit of 40 Mbytes,
- interface board for video, Ethernet coupling.

1.4 Software Configuration

1.4.1 Operating System

The SUN 4/330 operating system is UNIX which allows multiprocess applications, supports multi-windowing features and offers a wide range of data processing facilities.

1.4.2 Environment Software called MERCATOR (METHods and Realization for the Control of Attitude and Orbit of satellites)

MERCATOR is mainly composed of two different subsystems :

- real-time acquisition and first treatment of data coming from stations (Ariane telemetry, satellite telemetry, radar measurements and localization measurements). MERCATOR is compatible with HDLC protocol and asynchronous RS 232 transmissions,

- man machine interface : the environment software receives all the flight dynamics software necessary to perform the operational tasks described in 1.2.

MERCATOR allows flight dynamics experts to use within a short response time, various analysis tools in nominal cases as well as in degraded cases :

- real-time or batch mode data processing,
- synchronization of space mechanics modules,
- dynamic displays and plots,
- powerful control of input data and results.

Four people are sufficient to perform a complete positioning of a geostationary satellite and each member of the team can control all the operations even if there is a partial system failure.

Moreover, the workload for software implementation and integration is negligible and MERCATOR can be applied to a wide range of satellites.

The application software organization is depicted in Fig. 3 for geostationary missions. The telemetry and localization data are preprocessed after acquisition and stored in both raw and preprocessed forms. Each application module is autonomous and can be parameterized, operated and monitored through the multi-windowing facilities.

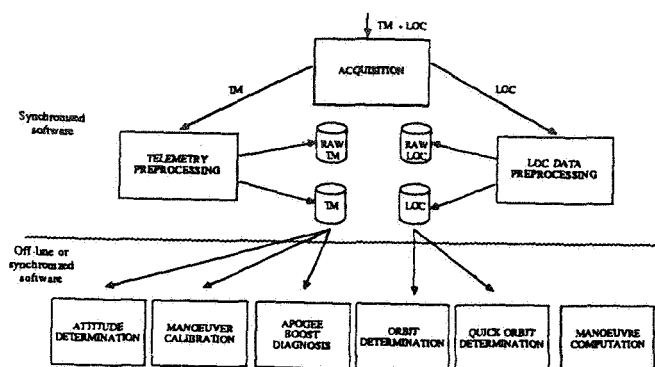


Fig. 3

2. THE HISPASAT MISSION

2.1 Background of HISPASAT LEOP

HISPASAT S. A., a Spanish company, has selected MATRA MARCONI SPACE (MMS) as prime contractor for the manufacturing and delivery of a space segment including :

- two telecommunication satellites, HISPASAT 1A and 1B delivered in orbit ; the two satellites are located at 30°W,
- a ground control system located in Arganda, near Madrid, including a satellite control center, Ku-band and S-band TTC stations, a payload monitoring center and specific facilities dedicated to LEOP operations.

HISPASAT 1A was launched on September 10, 1992, and HISPASAT 1B's launch is scheduled for April 1993.

The satellite structures are based on a EUROSTAR 2000 platform. They are compatible with a launch on Ariane 4 in upper or lower position.

The main characteristics of the satellites that impact on mission analysis are the following :

. Combined propulsion system :

HISPASAT uses a biquid propulsion system which supplies propellant to both the apogee engine and the attitude and orbit control thrusters. The apogee engine is used to bring the satellite from the elliptical transfer orbit delivered by Ariane 4 to the geostationary orbit by the mean of three apogee firings, thus utilizing several intermediate orbits.

. Spin stabilization :

During the transfer orbits, a 13 rpm spin rate is maintained, providing the satellite a passive stability particularly during the apogee firings. Attitude maneuvers give the spin axis an optimal direction before each firing.

. Three-axis stabilization :

After the third apogee firing, the satellite attitude is established in its normal three-axis configuration with the communication antennas directed towards earth.

For these two satellites, MMS has chosen CNES, the French National Space Agency, as the subcontractor to be responsible for LEOP activities including mainly :

- 2 GHz CNES/NASA ground stations network availability,
- flight dynamics center,
- operations control.

An integrated team, including CNES, MMS and HISPASAT personnel, was built up for the LEOP. At the FDC level, the four persons team is composed of two engineers from CNES and two from MMS.

A unique element of this mission is that all the HISPASAT LEOP operations are performed in Spain, using HISPASAT Satellite Control Center. The CNES Satellite Control Center is not employed; the Flight Dynamics Center was duplicated and MMS created a new main control room and a new satellite specialists room.

The Network Operation Center (NOC) remained in Toulouse and three specialized lines were installed between Toulouse and Madrid :

- . two data lines in hot redundancy for transmission of :
 - telemetry from the NOC to the SCC and FDC,
 - localization measurements from the NOC to the FDC,
 - telecommand from the SCC to the NOC,
- . one voice line which can be used for data transmission in case of double failure of the data lines.

The FDC mission preparation included the following phases :

- hardware and software installation in Madrid,
- integration with the other entities : ground stations, satellite control center, etc.,
- technical and operational qualification : simulation of nominal and degraded cases.

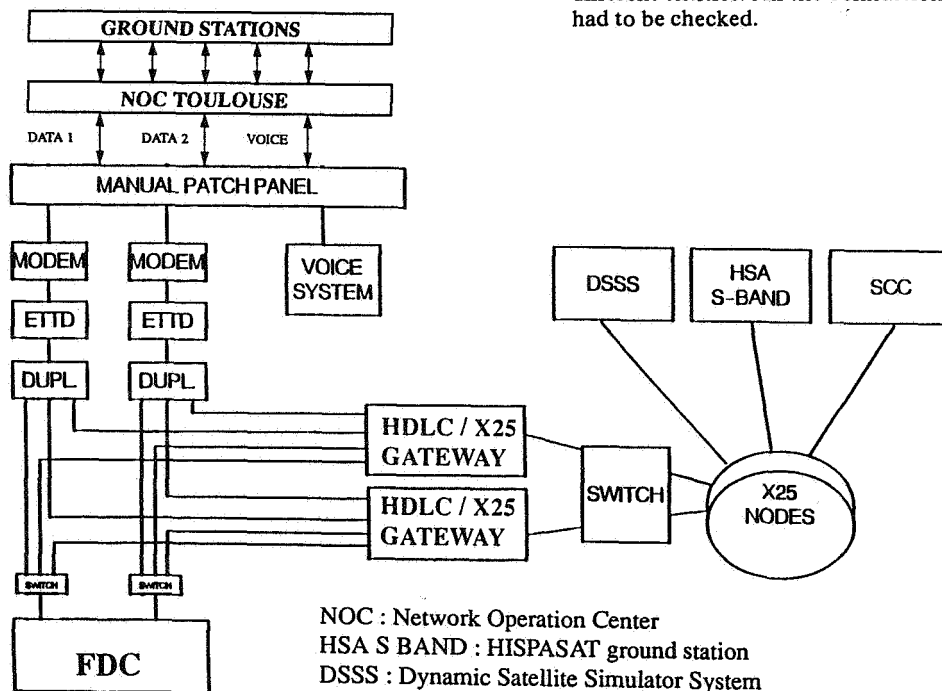


Fig. 4

2.2 Hardware and Software Installation

A reference hardware and software configuration in Toulouse that had been used for the Télécom 2 A and B satellites, which are very similar to HISPASAT 1A and 1B, was entirely duplicated at the Ground Control Center in Madrid.

Following specifications from FDC, MMS supplied and installed hardware and system software. Before the application software installation, FDC checked the following points :

- power supply,
- cabling,
- data transmission through the Ethernet network,
- video displays,
- laser printers,
- disks memory sharing,
- SUN operating system implementation,
- Transcript software implementation,
- HDLC software implementation,
- spare material,
- existing documentation,
- hardware maintenance organization.

Once this was completed, FDC installed the MERCATOR environment software and the flight dynamics software. To validate this essential part of the system, several test cases were defined for each flight dynamics module. They were run on the reference FDC in Toulouse and on the new FDC in Madrid. In every case, the same results had to be obtained.

2.3 Integration with the Other Entities

Once the installation of all the entities was completed, the technical validation of the global system began. It mainly concerned the data transmission between the different entities. All the connections described in Fig. 4 had to be checked.

Two data transmission protocols are used. The CNES LEOP system (NOC, FDC) uses the HDLC protocol, the HISPASAT Ground Control Center uses the X25 protocol. The two systems are linked by two gateways (for redundancy purposes) which enable data transmission in both directions.

For the FDC, three types of end-to-end lines had to be tested, corresponding to three different phases of the mission :

- line from the satellite simulator (DSSS) to the FDC through the gateway, for the operational qualification (see paragraph 2.4) when nominal and degraded configurations of the LEOP are simulated thanks to the telemetry provided by the satellite simulator,

- lines from the ground stations around the world to the Network Operation Center in Toulouse and from there to the FDC in Madrid through the two data flow lines. These lines allow the FDC to receive the satellite telemetry and the localization measurements during the main part of the LEOP,

- line directly from the HISPASAT S-band antenna in Madrid to the FDC through the gateway. This line is used at the end of the LEOP to receive the satellite telemetry when the satellite gets close to its station point.

2.4 Technical and Operational Qualification

This is one of the most important phases of the preparation of the LEOP mission. Its purpose is the training of the different teams to face nominal and degraded configurations that may happen during typical phases of the mission, i. e. spin axis reorientation, apogee engine firing, transition from spin stabilization to 3-axis stabilization, etc. The satellite behavior is duplicated by the satellite simulator which generates a telemetry flow. The degraded cases can be due as much from a satellite failure as to a ground system hardware failure. Beyond the training of the teams, these tests check the ability of the global system to find solutions and allow the responsibilities of each entity to be precisely organized.

2.5 Mission Preparation Effort

Globally for the FDC, the LEOP preparation lasted 18 months. Most of the time was spent in Toulouse, France, for :

- mission analysis,
- improvement of the flight dynamics software,
- preparation of the missions in Spain.

The missions in Spain lasted around 2 months which main part concerned the technical and operational qualification.

As the two FDC MMS engineers, since they had never participated to a LEOP mission before, these 18 months enabled them to learn the flight dynamics aspects from a theoretical and operational point of view.

3. RESULTS OF THE LEOP MISSION OF HISPASAT 1-A

3.1 Space Mechanics Results

HISPASAT 1-A was launched by an Ariane 44LP on the 10th of September in the evening at the opening of the launch window. The initial transfer orbit delivered by the launcher was very close to the nominal targeted orbit. The nominal strategy was successful, that is to say the three apogee engine firings happened as foreseen at the 4th, 8th and 11th apogees.

3.1.1 Attitude Determination

The objective of the attitude determination is to know the orientation of the spin axis in order to perform a slew maneuver to get the optimal apogee firing attitude.

For this, the FDC uses the data provided by the satellite Earth Sun Sensor (ESS) composed of two solar slits detecting the sun and two infra-red sensors detecting the earth. The required accuracy on attitude determination was 2 deg.

On the table below, one can see the accuracy obtained by comparing the determined attitude before the apogee maneuvers to the results of the calibration of the apogee maneuvers with the orbits before and after the firings.

	Apogee firing 1	Apogee firing 2	Apogee firing 3
Error on attitude (deg)	0.17	0.29	0.17

3.1.2 Orbit Determination

The high quality of localization measurements enabled FDC to consistently provide adjusted orbital parameters on time. During apogee firings 1 and 3, the satellite was within visibility of three stations which were configured to provide composite Doppler measurements enabling the FDC to perform real-time determination of the orbit during the maneuvers. The accuracy of the determined orbit at the end of the firings was better than 5 km on the semi-major axis.

3.1.3 Cost of the LEOP

Given the initial orbit provided by Ariane, the minimum cost (without any dispersion on the attitude determination, the orbit determination or the performances of the apogee engine) would have been 1 507.27 m/s. The speed increments really performed were the following ones :

- apogee firing 1 : 722.57 m/s,
- apogee firing 2 : 680.49 m/s,
- apogee firing 3 : 101.41 m/s,
- station acquisition maneuvers : 3.15 m/s.

The station acquisition maneuvers are the minor speed increments performed with the satellite thrusters after the third apogee firing to complete the station acquisition at 30° W longitude.

The total cost of the LEOP was then 1 507.62 m/s, which can be considered very good.

3.2 Operational Results

During HISPASAT 1-A LEOP, a team of only four people including two beginners operated MERCATOR successfully. From an operational point of view, the main performances of the system were :

. hardware security :

- the data acquisition was performed in hot redundancy on two workstations,
- three workstations were sufficient to perform the operations, with the fourth one used for hot redundancy,
- every monitoring display had a back-up during critical phases,
- there was a large margin for processing load (30 %) ;

. flexibility :

- each member of the team was able to perform any task and had, at any given time, a complete representation of the current positioning situation ;

. operational security :

- MERCATOR allowed flight dynamics experts to use various analysis tools and to perform cross checking of their results ;

. simplicity :

- each computation program uses standard interfaces and can be initialized easily due to powerful man machine interface,
- flight dynamics experts rapidly performed numerous runs in different modes to produce results with the best possible accuracy.

CONCLUSION

The LEOP mission of HISPASAT 1-A was very successful.

From the space mechanics point of view, this was due to a good initial orbit, the good behavior of the satellite, the quality of the attitude and orbit measurements, the accuracy of the attitude and orbit determinations, the good optimizations and calibrations of the apogee maneuvers and the reliability of MERCATOR software.

From the operational point of view, the MMS, CNES and HISPASAT integrated team functioned very well. Thanks to its flexibility, the integration of the FDC with the other entities was rapidly completed.

For upcoming missions abroad, the HISPASAT mission demonstrated the ease of moving the FDC and linking it to another satellite control center. This enables the customer to test its satellite control center during the preparation of the LEOP instead of waiting for the beginning of the station keeping.

REFERENCES

1. Belon B., Bergès J.-C., Campan G., Legendre P., "MERCATOR, A new ground system for orbit and attitude control", Second International Symposium on Spacecraft Flight Dynamics, Oct. 20-23, 1986.
2. Vielcanet P., Moran P., Secretan H., "MERCATOR, A distributed ground control system for geostationary satellite positioning, Ground data system for spacecraft control", ESA, International Symposium, June 26-29, 1990
3. Campan G., "Mise à poste INMARSAT-2/F1", IV Congrès Aérospatial Européen, Bases de lancement et infrastructures de contrôle des spatonefs, Paris, 13-16 mai 1991
4. Lazard B., "Launch and early operations phases for the dual launch of Télécom 2-A and INMARSAT-2/F3", 43rd Congress of the International Astronautical Federation, Aug. 28 - Sept. 5, 1992
5. Barbier C., de Boer F., "Kalman filtering applied to real-time monitoring of apogee maneuvers", Second International Symposium on Ground Data Systems for Space Mission Operations, Pasadena, California, Nov. 17-20, 1992,

SESSION I

OPERATIONS SUPPORT SERVICES

CO-CHAIRPERSONS

W. M. Edmonson, Bendix Field Engineering Corporation, USA
M. D. Sweetin, OAO Corporation, USA

Summary	643
W. M. Edmonson	
I.1 Intelligent Tutoring and Aiding in Satellite Ground Control	645
R. W. Chu, Honeywell Sensor & Systems Development Center, Minneapolis, Minnesota, USA; C. M. Mitchell, Center for Human-Machine Systems Research, Georgia Institute of Technology, Atlanta, Georgia, USA	
I.2 Operability Engineering in the Deep Space Network	651
B. Wilkinson, JPL, California Institute of Technology, Pasadena, California, USA	
I.3 The Application of Total Quality Management Principles to Spacecraft Mission Operations	655
M. Sweetin, OAO Corporation, Altadena, California, USA	
I.4 Mission Operations and Command Assurance: Instilling Quality into Flight Operations	661
L. L. Welz, M. M. Witkowski, K. J. Bruno, and S. S. Potts, JPL, California Institute of Technology, Pasadena, California, USA	
I.5 Operational Training for the Mission Operations at the Brazilian National Institute for Space Research (INPE)	667
P. Rozenfeld, INPE, São José dos Campos, Brazil	
I.6 Training: Plugging the Information Gaps	673
C. J. Scott, JPL, California Institute of Technology, Pasadena, California, USA	
I.7 The Columbus Logistics Support at the APMC: Requirements and Implementation Aspects	679
C. Canu, ASI, Rome, Italy; L. Battocchio, Alenia Spazio S.p.A., Turin, Italy; S. Masullo, Compagnia Italiana Servizi Tecnici (CISST) S.p.A., Space Division, Rome, Italy	
I.8 Automating a Human Factors Evaluation of Graphical User Interfaces for NASA Applications: An Update on CHIMES	685
J. Jiang, E. D. Murphy, and S. C. Bailin, CTA Incorporated, Rockville, Maryland, USA; W. F. Truskowski, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA	

I.9	Security Aspects of Space Operations Data	691
	S. Schmitz, TÜV Rheinland, Test Center for Software, Cologne, Germany	
I.10	Mission and Science Activity Scheduling Language	697
	L. G. Hull, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA	
I.11	Specifying Design Conservatism: Worst Case versus Probabilistic Analysis	703
	R. F. Miles, Jr., JPL, California Institute of Technology, Pasadena, California, USA	
I.12	SIMSAT: An Object Oriented Architecture for Real-Time Satellite Simulation	711
	A. P. Williams, Cray Systems, Space Division, Bristol, UK	

SUMMARY — SESSION I

W. M. EDMONSON
Bendix Field Engineering Corporation

The theme of this session was the diversity of operations support. The full spectrum of operations support services was covered, including design, operability engineering, training, total quality, commanding, security, logistics, scheduling, and automation.

ISSUES

- More emphasis is needed on preplanning — the right people at the right time every time.
- Software development, engineering, and operations all should be involved from the start.
- Prototyping must get closer to actual operations, very early in the development process.

OBSERVATIONS

- Both the co-chairpersons are from operational backgrounds; thus, the challenge of proper paper selection was simplified.
- A good cross-section of papers was selected, including three presented by representatives from overseas organizations.

- Two of the scheduled papers were not presented.
- All the presenters were well versed in their specialties and obviously very dedicated.
- Most presenters stayed within their allotted time periods, but it was obvious that with their enthusiasm, they could have continued much longer.
- There was a very positive exchange of information between presenters and also between the presenters and the audience, with several of these resulting in after-session follow-on meetings.
- Regardless of the field of experience, it was clear that the total dedication to achieving flawless spaceflight operations is alive and well.

N94-23929

INTELLIGENT TUTORING AND AIDING IN SATELLITE GROUND CONTROL

Rose W. Chu **

Sensor & Systems Development Center
Honeywell MN65-2300
3660 Technology Dr.
Minneapolis, MN 55418

rose@ssdc.honeywell.com
(612) 951-7471

Christine M. Mitchell

Center for Human-Machine Systems Research
School of Industrial and Systems Engineering
Georgia Institute of Technology
Atlanta, GA 30332-205

cm@chmsr.gatech.edu
(205) 894-4321

ABSTRACT

In supervisory control system such as satellite ground control, there is a need for *human-centered automation* where the focus is to understand and enhance the human-system interaction experience in the complex task environment. Operator support in the form of offline intelligent tutoring and online intelligent aiding is one approach towards this effort. The tutor/aid paradigm is proposed here as an desing approach that integrates the two aspect of operator support in one system for technically oriented adults in complex domains. This paper also presents GT-VITA, a proof-of-concept graphical, interactive intelligent tutoring system that is a first attempt to illustrate the tutoring aspect of the tutor/aid paradigm in the domain of satellite ground control. Evaluation on GT-VITA was conducted with NASA personnel with very positive results. GT-VITA is presented being fielded as it is at Goddard Space Flight Center.

Keyword: Intelligent tutoring systems, operator support, technical training, supervisory control.

1. INTRODUCTION

In very highly automated systems such as satellite ground control and process control, the human operator is involved in supervisory control task -- monitoring system states and compensating for system failures (Ref. 10, Ref. 13). On one hand, advances in computer technology over the years had contributed to the efficiency and productivity of automated systems in terms of high-performance equipment and innovative automation control strategies/algorithms. On the other hand, it is clear that the operator's changing role in

automation has also resulted in a new set of problems with prevailing issues in reliability and safety. This is evident in tragedies such as the three-mile island accident. Consequently, there is a need for *human-centered automation* where the focus is to explore technology not just for better automation, but for understanding and enhancing the human-system interaction experience in the complex task environment resulted from automation.

To reduce task complexity, Leplat (Ref. 8) suggests either changing the operator through training or changing the task through redesign. At NASA Goddard Space Flight Center, existing training methods to prepare novice for satellite ground control at the Mission Operations Division include classroom instruction, quizzes, volumes of reading materials, and on-the-job training. Unfortunately these methods are inadequate by themselves in meeting new training requirements imposed by the nature of supervisory control task and the complexity of the task domain. Operators require high-level skills beyond rote memory of procedures and rules to manage trouble when trouble arises, and to copy with unanticipated variabilities in system behavior. In addition, for on-the-job training, there is the problem of knowledge attrition that results as new generations of operators are being trained over time.

Hence the primary motivation for the research work reported in this paper is to explore the design of an intelligent tutoring system (ITS) for training novice operators in satellite ground control.

For the next generation of mission control systems, researchers and engineers at Goddard are involved in changing aspects of the satellite

** This work was completed in September 91 as part of Rose Chu's doctoral program at Georgia Tech.

control task through redesign of existing systems, capitalizing on recent technological advances. While some existing problems may be come obsolete in the new systems, others may still arise because of the complexity of the application in an even more high-tech environment. Thus, there is still a need to address the issue of online operator support such as intelligent aiding systems that may potentially facilitate an operator in performing supervisory control activities. More importantly, many existing systems that have been in place for many years are often not amenable to redesign except potentially through distributed operator support tools and aiding systems.

Hence, the research presented here also suggests a design paradigm that considers intelligent training and aiding as two ends in the spectrum of operator support.

2. THE TUTOR/AID PARADIGM

Conventionally, research in training has been fielded separately from research in aiding. We propose the tutor/aid paradigm in which an intelligent tutoring system (ITS) evolves from a tutor to an operator's assistant as the novice operator's expertise increases. The tutor/aid paradigm integrates the issues of computer-based training and real-time aiding in complex dynamic systems. We believe that a system developed to function both as a tutor and an aid is more cost-effective over time, and is more motivating for operators to learn, to use and to trust. The goal is to design a tutor that not only compensates for a novice operator's deficiencies in knowledge and skills, but also prepares the operator to use the tutor as an assistant after training.

The tutor/aid paradigm proposes characteristics of an integrated intelligent tutoring and aiding system. The heart of which is a pedagogy structure that address "how and when and what" to teach to the novice operator of complex systems that eventually uses the tutor as an aid. Before the theory of the tutor/aid paradigm was defined, we investigated the recent developments in the field of intelligent tutoring systems. While research in intelligent aiding in complex domains has been emerging

with existing and successful methodologies, it is not the case with intelligent tutoring systems.

2.1 Academic vs. Technical Tutors

A survey of most of the ITS projects shows that the application domains prevalent in the field have been academic subjects matters such as geography, mathematics and computer programming. The application of ITS for training in engineering domains such as electronics and power plants has begun only in recent years. The distinction between academic ITSs and technical training ITSs is crucial because methods and concepts that work in academic tutors may or may not work in technical tutors because of the nature of the target domain and the target student that each class of tutors is generally designed for.

Consider the world of algebra and the world of steam propulsion plants. A tutor preparing a student for the former is very different from one for the latter due to the difference in knowledge and skills requirements imposed by the domain. From the vantage point of an individual performing some tasks in a world, a steam propulsion plant is more complex than algebra in dimensions of dynamism, the number and extensiveness of interacting parts, uncertainty and risk (Ref. 14).

Academic tutors differ from training tutors not only in "what to teach", but also in "who" they are teaching. In industry, novice operators who are mature and technically oriented adults are highly motivated to learn and be trained specifically because of their vested interest in job performance, and in most cases, because their job mandates them to do so. In contrast, whether the students are first-graders or college freshmen, in an academic environment, students often learn for the sake of learning and in preparation for future vocations. Thus, at the time, they usually do not have a job at stake.

2.2 Dimensions of an ITS

From a design perspective, an ITS can be addressed from three aspects: communication, instruction and knowledge (Ref. 1). The communication dimension concerns the "interactivity" between the student and the tutor via the interface. The interactivity of recent

learning environments has been greatly facilitated by the advancement in computer graphics and interaction techniques.

The instructional dimension concerns the representation of teaching knowledge and of the student so that individualized instruction can be imparted. Of the three, the instructional dimension is the most difficult and least developed. Most ITSs to date embody some pedagogical philosophy but lack a structure that explicitly encodes the teaching knowledge. In addition, many student models to date are simply good recordkeeping of student actions.

The knowledge dimension concerns the representation and presentation of the target domain. This dimension is probably the most developed, benefiting from the research and experiences in artificial intelligence.

The representation issue is especially critical in *knowledge-rich* domains, where the tutor teaches the conscious and accurate use of knowledge in a high-level problem solving task such as electronic troubleshooting (Ref. 4). Whereas in *high-performance* domains such as air intercept control, the focus is on speed and automaticity: the tutor trains a student to perform some skills as effortlessly and reliably as possible (Ref. 4, Ref. 11). Most ITS research to date has emphasized knowledge-rich domains or knowledge-rich components of the domain such as troubleshooting and fault diagnosis.

2.3 Scope of the Tutor/Aid Paradigm

In summary, based on the survey and analysis on current developments in ITS, the tutor/aid paradigm is an attempt to address some research needs in the field. First, the current work emphasizes the design of an ITS for technically-oriented adults in a complex and dynamic environment. Second, the focus is not just on training some knowledge component aspect of the task, but on training for the successful operations of supervisory control activities in realistic, time-constraint scenarios. Third, the tutor/aid paradigm proposes a pedagogy structure that explicitly defines the training process of the integrated tutor and aid.

Fourth, instead of designing an ITS from scratch, the tutor/aid paradigm capitalizes on the

emerging research in operator associates. Specifically, the current research enhances a validated operator's associate architecture that is based on a prescriptive model of human performance for supervisory control to include tutoring capabilities. The Operator Function Model (OFM, Ref. 9), in effect, provides the intelligence for the ITS by defining the target training state towards which a novice operator should be trained.

The remaining of the paper briefly discusses GT-VITA (Georgia Tech-Visual Inspectable Tutor and Assistant), a proof-of-concept ITS that is an initial attempt to illustrate and validate the tutoring aspect of the tutor/aid paradigm. Details of the tutor/aid paradigm can be found in Ref. 3. The aiding aspect of the paradigm is beyond the scope of this research but is addressed and complimented by a companion project on cooperative problem solving (Ref. 6).

3. FROM GT-POCC TO GT-VITA

The GT-VITA project can be viewed as a direct response to the training needs and problems of existing systems at the Mission Operations Division of GSFC.

A necessary first step of the project is to conduct extensive analysis on the NASA data-information system and the operator's knowledge and task requirements within the Payload Operations Control Center (POCC). Based on this analysis, we developed an interactive, real time simulation system called GT-POCC (Georgia Tech-POCC, Ref. 2).

3.1 GT-VITA Interface

GT-POCC provides the simulated task environment upon which GT-VITA is built up. GT-VITA is a graphical, interactive, real-time intelligent tutoring system. GT-VITA provides a graphical and animated representation of the NASA data-information system that allows a student to visualize system components, behavior and their relations at a various level of details. The graphical interface was developed in direct response to some concerns that were identified in our knowledge analysis. Operators at the POCC do not have a broad, top-down perspective of the NASA system (Ref. 7). Such a perspective has been argued to be crucial for

appropriate learning of and reasoning about a system's normal behavior and its reactions to different types of failures (Ref. 5).

The graphic interface of GT-VITA is not only used during training, it is intended to function as support for real-time operations after training, as illustrated in the companion project GT-MOCA (Georgia Tech-Missions Operations Cooperative Assistant, Ref. 6). Figure 1 is the NASA system graphic display, one of many used both for GT-VITA and GT-MOCA.

GT-POCC and GT-VITA have been implemented in C++ on Sun SparcStation platforms. The user interface have been created with Transportable Applications Environment Plus (TAE Plus, Ref. 12), an interface builder developed at GSFC.

3.2 GT-VITA Tutoring Capabilities

The tutoring capabilities of GT-VITA are captured in a pedagogy module that defines the instructional goals and strategies of the tutor in the form of eight lesson types. Figures 2 shows the pedagogy network as proposed by the tutor/aid paradigm. At present, GT-VITA implements seven of the eight lessons in three phases: declarative, procedural and practice. Right now, GT-VITA is not yet capable of the dynamic transition between lessons that the paradigm ultimately proposes.

A typical training session in the GT-POCC domain is as follows: In the declarative phase of training, the student learns about the various NASA components such as spacecraft subsystems and ground computers, followed by the behavior of the NASA system in terms dynamic data flow between components. During this phase, the student also explores the NASA system both from the visual perspective as well as from the perspective of the tutor's internal hierarchical representation of the system.

In the procedural phase, a student learns about the various satellite ground control activities first via tutor's demonstrations, followed by tutor's instruction to perform the activities.

Finally, in the practice phase, a student has the opportunities to apply previous learned

declarative and procedural knowledge in real-time scenarios. The consensus among POCC operators is that all the concepts and tasks are not difficult by themselves. It is when they have to be applied and coordinated under limited time and dynamic conditions. In the first type of practice lesson, the tutor provides direct feedback for untimely or erroneous actions. In the second type of practice lesson, the tutor only intervenes during critical phases of the real-time scenario.

4. GT-VITA EVALUATION AND RESULTS

An evaluation of GT-VITA was conducted on site at GSFC, in June 1991. Sixteen NASA personnel volunteered six to nine hours of their time to participate in the evaluation. The subjects included software testing engineers, administrators, spacecraft specialists and computer operators. The subjects varied in their experience with computer technology (e.g., using the mouse). They also varied in their general knowledge of the NASA system and the duties of a POCC operator. However, none of the subjects had on-the-job experience as a POCC operator. All subjects were trained on GT-VITA with all seven lessons. The evaluation procedure consisted of subjective questionnaires, and objective performance measures.

In general, results show that GT-VITA was perceived favorably by all the subjects in terms of what they learned. All subjects agreed that GT-VITA would be useful for training novice POCC operators.

An analysis of the performance data collected for the practice lessons showed that regardless of the subjects' differing background, all subjects performed comparably and even committed consistent errors in the practice lessons. These results can be attributed to the flexibility and adaptability afforded by GT-VITA.

Overall, the response of other NASA training personnel and managers have been very positive. Some have commented that GT-VITA could save them several months of training time. GT-VITA is currently being fielded as it is at NASA, and also modified for the next generation T-POCC (Transportable POCC) applications. Eventually, the two different

versions will be integrated with the overall training program for novice satellite ground control operators.

In conclusion, the GT-VITA project represents a first attempt in understanding and addressing issues in computer support systems for operators of complex dynamic systems. Many research avenues need to be conducted further to validate the full impact of the tutor/aid paradigm. The tutor/aid transition is being investigated at Georgia Tech. Other enhancements to GT-VITA such as student modeling and curriculum transition are being explored at Honeywell.

5. ACKNOWLEDGMENTS

We gratefully acknowledge the assistance provided by the Mission Operations Division at NASA Goddard Space Flight Center, especially the NASA personnel who volunteered their time to participate in the evaluation of GT-VITA. This research was sponsored by NASA Goddard Space Flight Center under contract NAS5-28575 (Walt Truszkowski, technical monitor).

6. REFERENCES

1. Burns, H. and Parlett, J.W. (1991). The evolution of intelligent tutoring systems: Dimensions of design. In H. Burns, J.W. Parlett, and C.L. Redfield (Eds.), *Intelligent tutoring systems: Evolution in design*. Hillsdale, NJ: Lawrence Erlbaum Associates.
2. Chu, R.W., Jones, P.M., and Mitchell, C.M. (1990). The Georgia Tech Payload Operations Control Center simulation: Design and implementation in C++. *Proceedings of the SCS Multiconference on Object-Oriented Simulation*, January 23-25, Anaheim, CA.
3. Chu, R.W. (1991). The tutor/aid paradigm: Design of intelligent tutoring systems for operators of supervisory control systems. Ph.D. Dissertation, CHMSR, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA.
4. Fink, P.K. (1991). The role of domain knowledge in the design of an intelligent tutoring system. In H. Burns, J.W. Parlett, and C.L. Redfield (Eds.), *Intelligent tutoring systems: Evolution in design*. Hillsdale, NJ: Lawrence Erlbaum Associates.
5. Goodstein, L.P., Andersen, H.B., and Olsen, S.E. (Eds.) (1988). *Tasks, errors and mental models*. London: Taylor & Francis.
6. Jones, P.M. (1991). Human-computer cooperative problem solving in supervisory control. Ph.D. Dissertation, CHMSR, School of Industrial and Systems Engineering, Georgia Institute of Technology, Atlanta, GA.
7. Jones, P. M., Chu, R. W. and Mitchell, C. M. (1990). Knowledge requirements for the flight operations team analyst in the Goddard Space Flight Center Advanced POCC. CHMSR, School of ISyE, Georgia Institute of Technology, Atlanta, GA, unpublished.
8. Leplat, J. (1988). Task complexity in work situations. In L.P. Goodstein, H.B. Andersen, and S.E. Olsen (Eds.), *Tasks, errors and mental models*. London: Taylor & Francis.
9. Mitchell, C.M. (1987). GT-MSOCC: A domain for research on human-computer interaction and decision aiding in supervisory control systems. *IEEE Transactions on Systems, Man, and Cybernetics*, 17(4), 553-572.
10. Rasmussen, J. (1986). *Information processing and human-machine interactions: An approach to cognitive engineering*. New York: North-Holland.
11. Regian, J.W. (1991). Representing and teaching high performance tasks within intelligent tutoring systems. In H. Burns, J.W. Parlett, and C.L. Redfield (Eds.), *Intelligent tutoring systems: Evolution in design*. Lawrence Erlbaum Associates, Hillsdale, NJ.
12. TAE Plus (1990). Overview Version 4.1, NASA Goddard Space Flight Center, January, 1990.
13. Wickens, C.D. (1984). *Engineering psychology and human performance*. Columbus, OH: Merrill.
14. Woods, D.D. (1988). Coping with complexity: The psychology of human behaviour in complex systems. In L.P. Goodstein, H.B. Andersen, and S.E. Olsen (Eds.), *Tasks, errors and mental models*. London: Taylor & Francis.

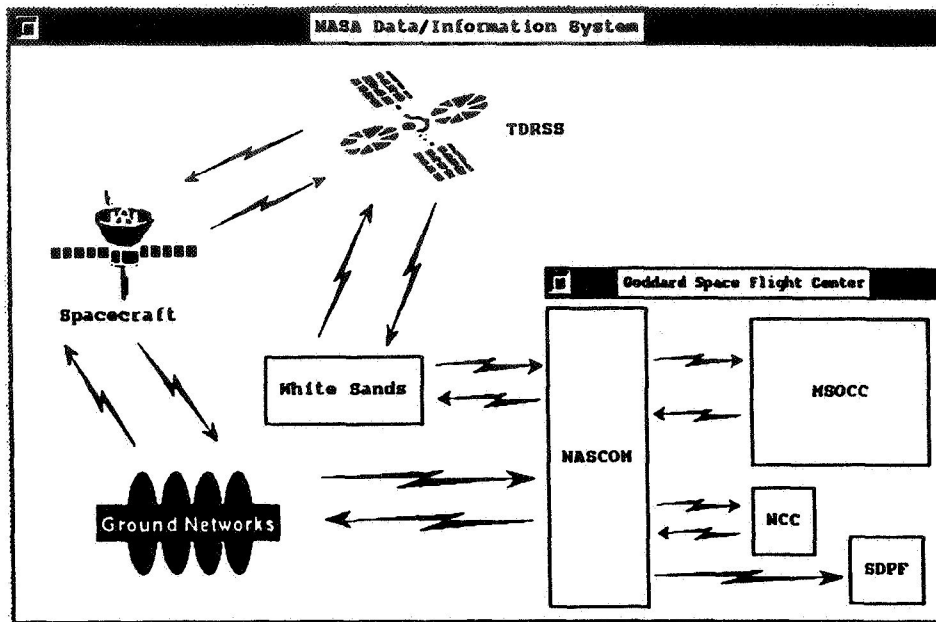


Figure 1. The NASA System Graphic Display of GT-VITA

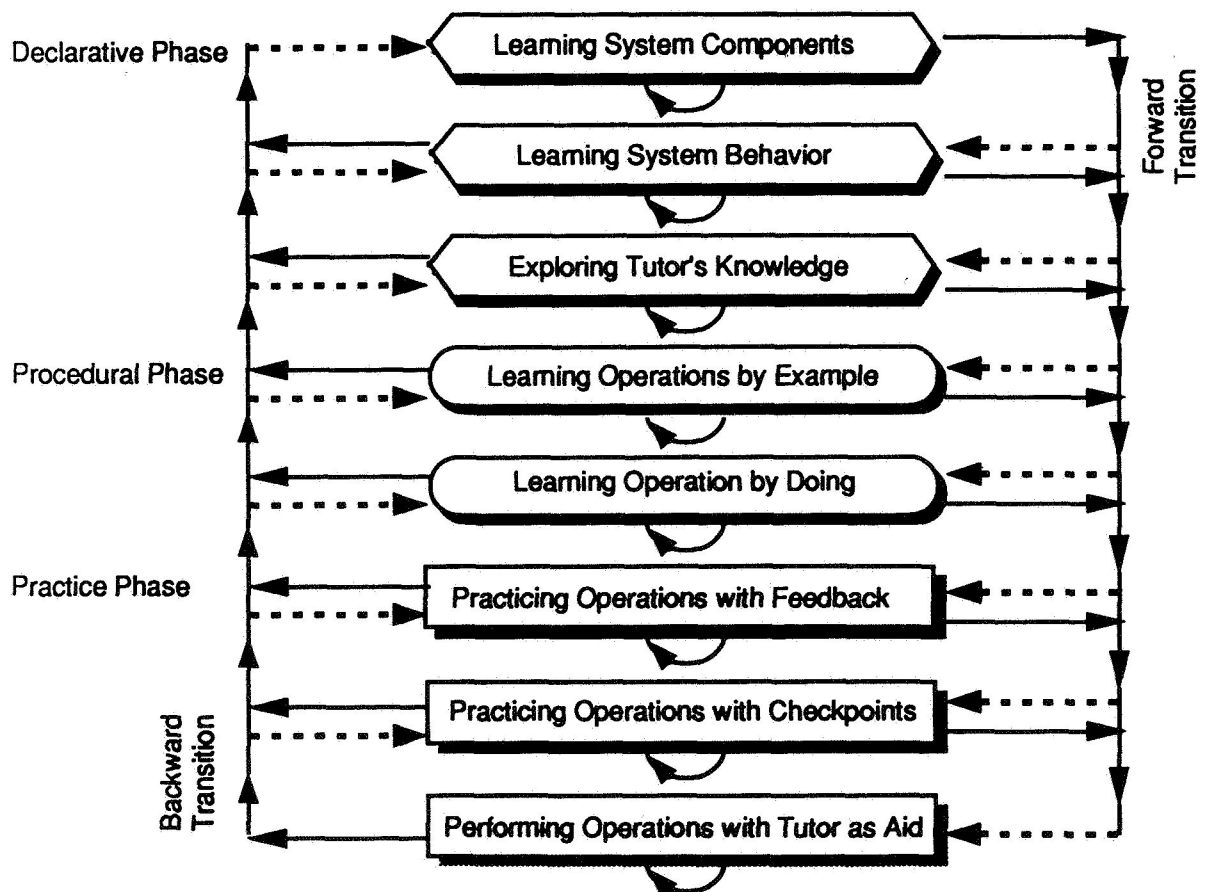


Figure 2. The Pedagogy Network of the Tutor/Aid Paradigm

OPERABILITY ENGINEERING IN THE DEEP SPACE NETWORK

Belinda Wilkinson

N 9 4 - 2 3 9 3 0

Jet Propulsion Laboratory, California Institute of Technology
Pasadena, California/USA

ABSTRACT

Many operability problems exist at the three Deep Space Communications Complexes (DSCC's) of the Deep Space Network (DSN). Four years ago, the position of DSN Operability Engineer was created to provide the opportunity for someone to take a system-level approach to solving these problems. Since that time, a process has been developed for establishing communication between operations personnel and development engineers and for enforcing user interface standards in software designed for the DSCC's. Plans are for the participation of operations personnel in the product life-cycle to expand in the future.

Key Words: operability, communication, standards, documentation, user interface

1. BACKGROUND

The Deep Space Network (DSN) is a space-to-ground telecommunications network operated by the Jet Propulsion Laboratory (JPL) for NASA. It is used for supporting deep-space missions, high Earth-orbiting flight projects, and radio astronomy experiments. It consists of multiple facilities, including a central Network Operations Control Center (NOCC) at JPL, the Central Communications Terminal (CCT) at JPL, and three Deep Space Communications Complexes (DSCC's), located in Goldstone, California; Madrid, Spain; and Canberra, Australia.

There are many people within the JPL community with a legitimate interest in the requirements, design, and implementation of DSN subsystems. Since JPL has a matrix management organizational structure consisting of program offices and technical divisions, the responsibilities for different phases of the product life-cycle cut across organizational boundaries. Also, since DSN operations facilities are worldwide, there are

several different organizations in different physical locations that are part of the operations structure. Communication problems can develop as a result of this distribution of personnel across organizational boundaries and over many physical locations.

Four years ago, the DSN created the position of the DSN Operability Engineer, with the charter to ease the operational complexity of the network. This position requires someone with a background in both software engineering and human factors engineering, as well as considerable management experience.

This paper describes the process that has taken place since I was appointed to the position of the DSN Operability Engineer.

2. OPERATOR POINT-OF-VIEW

One of the first steps I took after receiving the new assignment was to visit the DSCC sites in California, Spain, and Australia. This provided me an opportunity to meet operations personnel and learn about their concerns and problems. I spent considerable time observing day-to-day operations and talking with operators. Other JPL engineers had done this, but as one DSCC operator later commented, "That was the first time that someone actually wrote down the comments that we made." In general, operators felt that developers never listened to them and never incorporated solutions to their complaints in new software releases. It was also the operators' perception that the DSN is a collection of independent subsystems rather than a cohesive system.

Actually, developers at JPL were very aware that operators were not happy, but did not always understand the specific areas of concern or have the expertise to solve the problems. Even when problems were understood, no one before had ever had the responsibility to collect the comments and provide a system-level approach to responding to them. That was the service that I could perform.

3. THE PROBLEM DEFINED

Many operability problems were discovered during these discussions with operators. As an example, at the DSCC's there are over 1500 commands that the operator must use to control the system. Since the current Monitor and Control subsystem only provides a command-line interface, the operator must memorize the 1500 commands. Many of the commands, particularly in the area of subsystem configuration, are not necessary, since the appropriate information could be accessed by the software from files. Another problem is that similar functions in different subsystems are often initiated by different commands: for example, the command to perform subsystem configuration is variously CNF, CONF, or CFG. These problems have developed because within each operational site are numerous subsystems that were designed, developed, and upgraded at different points in time. Traditionally, there has been little communication among the engineers responsible for developing the various subsystems; thus, a set of subsystems has been created that is not always cohesive in the operational environment.

There were many other operability problems. Owing to insufficient automation, information was frequently communicated on pieces of paper, thus requiring extensive key-in's on the part of the operator. The displays used for monitoring the status of the DSN were far too crowded, and the out-of-date technology prohibited true hierarchies of displays. Log displays were congested with too many messages, thus causing important messages to be lost in the heavy flow of traffic. The monitor and control equipment was antiquated, and monitors were poorly placed from an ergonomic point of view. Some of the monitors were so old that the clarity of the display presented was very poor, causing difficulty in reading the data.

Another operability problem was the lack of common displays and acronyms among the operational facilities: the DSCC's, the NOCC, and the CCT. The operators at the facilities are constantly in voice communication, and difficulties arise because of a lack of standards in presenting information (e.g., the unit of measure for a particular data item). This problem extends beyond the DSN boundaries to the DSN interface with the Multi-mission Ground Data System (MGDS) in the JPL Space Flight Operations Facility, from which another version of the data may be given to an operator.

4. COMMUNICATION

Three years ago, several major tasks were starting up that were designed to enhance the DSCC's with the delivery of many upgraded subsystems. One goal of these tasks was to improve the operability of these subsystems, although many of the operability problems cannot be resolved until a new Monitor and Control subsystem is provided. Although the Monitor and Control subsystem was not scheduled for replacement, the operability goal was set to eliminate as many problems as possible.

One major goal I had was to increase the level of communication between the operations personnel at the DSCC's and the development engineers at JPL. At that time, no formal mechanism existed for communication, and any communication that occurred did so on an ad hoc basis. Attempts at effective communication were made by using such techniques as videoconferences, overseas phone calls, and operability meetings at JPL. Pivotal in the success that was achieved was the appointment of a person at each DSCC to serve as the Operability Interface Engineer for that DSCC. That person is responsible for talking with the various DSCC crews and compiling the operator input into one integrated viewpoint for that DSCC.

The forum for communication that has proven to be the most successful is to send information electronically and then to have follow-up telephone conferences as needed. The major problem with this method of communication is one of time constraints. The personnel at the DSCC's are very busy and cannot always read and critique a document immediately upon receiving it. Of course, the JPL development engineers generally need to have answers very quickly if they are to remain on schedule. We have not solved the resulting conflict effectively.

Early on, videoconferences were tried, but these did not prove to be very successful forums for communication. The data rates of the video-conference system we used were so low that all viewgraphs had to be sent ahead of time. Since it took several minutes to paint a screen, sending anything in real time was not practical. Although time zone differences exist, from California to Australia to Spain, these differences were not a major problem since the involved people showed a willingness to come in during the night, if necessary, in order to participate.

Face-to-face communication with representatives from the DSCC's has been possible because, as part of the tasks, each DSCC provides one person to work at JPL as part of an Independent Test Group (ITG). Operability meetings have been held with the ITG members and JPL development engineers participating. These groups have been very useful, but are limited in their decision-making ability in that the ITG member is not always the Operability Interface Engineer for that DSCC and thus may not be able to speak officially for the DSCC.

In alternate years, the Operability Interface Engineers from the DSCC's come to Pasadena for an Operability/Monitor and Control workshop that generally lasts for one week. The topics discussed are quite varied, from anomalies in the current subsystems to creative thinking concerning the future. These meetings have been very useful in providing an opportunity to exchange ideas and make decisions.

5. DOCUMENTATION

Another opportunity to improve operability was to enforce the rule to design and document the user interface early in the design process, rather than to allow the development engineers to design the user interface during implementation and to document whatever user interface was delivered. This was accomplished by holding a series of rigorous reviews of the software operators manuals with the participation of operations personnel during the time frame of the high-level design. The challenge here is to follow through and make sure that the development team does not change the interface during implementation. The ITG team members test for this as part of their function.

Although user interface standards already existed in the DSN, the standards had never been successfully enforced. Enforcement has now been accomplished via the early review of the software operators manuals by operations personnel and the signature approval of the manuals by the Operability Engineer. The early review of the manuals has also increased the usability of the subsystems.

To improve the usability of documentation for operations personnel, a new format for the software operators manuals was developed with the participation of the users. The new format is designed to present an efficient, real-time

document, with certain sections designed for quick reference and other sections for more detailed study. A section on messages and suggested operator responses is included. The manuals are in a landscape format, so that a picture of a display and the written description of that display are available on back-to-front pages for easy viewing. Since work space at the operator consoles is limited, the manuals are currently in the process of being printed on smaller (8.5 inches by 5.5 inches) paper. They will be put in binders for ease of use. Also, a quick-reference manual containing information on all subsystems is being developed.

6. SUCCESSES AND FAILURES

One evidence of success is that development engineers are more conscious of operability issues these days. They hold meetings on the issues, ask questions, and find they are rewarded by talking with operations personnel. In general, there is a heightened awareness of operability issues. The following is a description of some of the specific successes and failures.

There is far more communication these days between operations personnel and JPL engineers. Currently, a proposal is being developed that will formalize the methodology for the participation of operations personnel in the entire development life-cycle of all subsystems in the DSN, from requirements through implementation. We have evolved from almost no participation four years ago, to participation in the design phase, to participation in all phases.

An operability module has been written for the DSN baseline requirements document that will govern all new subsystems. A preliminary requirements document for a new Monitor and Control subsystem at the DSCC's is being reviewed. This document contains many requirements designed to remove operational complexity from the DSCC's.

Other successes include a set of standard commands and display acronyms, online help screens, and standard system-administration procedures. New subsystems provide for configuration with one command by having the software access parameters from files sent electronically from JPL, rather than requiring the operator always to enter the parameters manually. Also, operations personnel were allowed to participate early enough in the

design process that frequently their input was incorporated into the final products. When the design could not accommodate their input, at least the operators were able to understand the reasons and learn the period of time when the product would not have the feature(s) they wanted.

The new subsystems use a common software program set providing standardization of many user interface features. Since the code is being written by one group, standardization is assured.

In the Monitor and Control subsystem, operations personnel have contributed very instrumentally to the requirements for the various releases. Operators had long been unhappy with the position of one of the four monitors found on each operator workstation. Discussions were held with the DSCC's, and all new workstations have the monitor positioned in a more ergonomically favorable location. Also, the worst monitors have been replaced with newer ones to improve readability.

The major failure, so far, involved one particular subsystem. Even though the subsystem development team talked extensively with the personnel at one DSCC, they did not write a software operator manual early in the design process and did not implement features agreed to in the meetings. Very few operability items were incorporated in the first release of that subsystem, but subsequent releases are addressing the items that were asked for by the DSCC personnel.

7. FUTURE

There are still major improvements to be made.

The following items are the ones to be implemented over the next couple of years.

In order for operations personnel to evaluate proposed designs, an operations scenario needs to be developed for each subsystem and activity, starting at requirements generation and being further refined at each stage in the development life-cycle. A standard for this operations scenario needs to be developed.

A more extensive data base of acronyms and units of measure needs to be developed to be used throughout the DSN--from the DSCC's, to the CCT, to the NOCC. Common displays need to be developed for utilization by all DSN facilities and by the Multimission Ground Data System in the JPL Space Flight Operations Facility.

New standards must be developed for graphical user interfaces, and a user interface presentation must be developed that allows the DSN subsystems to be viewed as a system by the operator, and not just as an ad hoc collection of program sets.

With all of these changes, the DSN will be able to continue doing an excellent job of tracking and data acquisition, and the operational environment will continue to be improved for the DSN operators.

8. ACKNOWLEDGMENT

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration (NASA).

THE APPLICATION OF TOTAL QUALITY MANAGEMENT PRINCIPLES TO SPACECRAFT MISSION OPERATIONS

N94-23931

Maury Sweetin
OAO Corporation
Altadena, California

ABSTRACT

By now, the philosophies of Total Quality Management have had an impact on every aspect of American industrial life. The trailblazing work of Messrs Deming, Juran, and Crosby, first implemented in Japan, has "remigrated" across the Pacific and now plays a growing role in America's Management culture. While initially considered suited only for a manufacturing environment, TQM has moved rapidly into the "service" areas of offices, sales forces, and even fast-food restaurants.

The next logical step has also been taken - TQM has found its way into virtually all departments of the Federal Government, including NASA. Because of this widespread success, it seems fair to ask whether this new discipline is directly applicable to the profession of Spacecraft Operations. The results of quality emphasis on our contract at JPL provide strong support for Total Quality Management as a useful tool in doing Spacecraft Operations.

Key Words: Total Quality; Mission Operations; Continuous Improvement; Spacecraft Operations; Process Improvement

1. INTRODUCTION

The purpose of this paper is to explore whether the unique aspects of Spacecraft Operations make it a logical candidate for the application of Total Quality Principles. We all are aware of the growing percentage of Mission Life costs which now must be allocated to the Operations Segment. Can TQM provide an answer to this and other operations-unique problems?

We can point to some direct experience to support the case for Operations TQM. In 1988, OAO Corporation was awarded the MACS contract, to support the Institutional Computing and Mission Operations Division's operational needs. As Mission

Operations Manager on this contract, I have been involved for nearly five years in providing Operations Manpower to the various Spacecraft Support activities of the Division. From the outset, this contract encouraged us to look for better ways of doing business, by making a significant part of our profit subject to continuous improvement. The intervening years have provided us with a very positive experience, and forms the basis for our belief that TQM in Operations is here to stay.

2. A PROGRAM BY ANY OTHER NAME..

Our own experience in Continuous Improvement, under the name Productivity Improvement and Quality Enhancement (PIQE), preceeded the widespread popularity of TQM by a couple of years. However, the goals were much the same, namely, to look into the Processes which make up our operations, and find ways to improve them. The added benefit of TQM, once it came into our field of view, was to give structure to our search for efficiency. In retrospect, then, we were doing basic TQM in 1988, but have benefitted since by the broader vision offered by the formal TQM program.

3. DOES TQM PASS THE TEST FOR GENERAL SPACECRAFT OPERATIONS?

It is fair for Managers at other locations and with different problems to ask whether this experience can be "transplanted", or whether our positive experience is due to unique circumstances. In order to answer this question, I have decided to choose a popular summary of the TQM philosophy, and to address each of its points as to applicability to a Spaceops environment. Where applicable, our experience at JPL will be included to make the case.

Selecting a single source for defining "TQM" is at this point difficult. By now, thousands of articles and books have been written on the subject, many training courses have been taught, and Consultants ready to further educate us abound. Yet, for our purposes, I believe we can do no better than to use as a reference one of the early "Prophets" of Total Quality, namely W.E. Deming.(Ref 1) Dr. Deming is by now some

ninety years young, and was a prominent figure in the revitalization of Japan during the early 1950's. Although his complete message on quality transformation has filled volumes, he has distilled its essence into what he calls his Fourteen Points. It is against these standards that we will consider TQM viability to Spacecraft Operations.

4. DEMING'S FOURTEEN POINTS

4.1. Point 1. Create Constancy of Purpose. (The first job of an organization is "staying in business" by innovating and constant improvement.) That Spacecraft Operations would actually go out of business may seem to be a remote likelihood. Most of it is funded directly or indirectly by Governments, which have supported space exploration for decades. Yet, we know that, over even longer term history, scarce resources will be allocated according to perceived need and payoff. Thus, we must be sensitive to the cost-benefit decisions which occur each year at budget time, to make sure that the taxpayer will continue to see a positive value to our efforts in the exploration and exploitation of space. We must grasp new technologies, abandon inefficient methods, and always be looking for better ways of doing business. As Operations costs increase, Managers must use new paradigms for looking at workload. Can the next job be integrated into the current workforce? Are we using the most efficient technology? How can the process be improved?

On the MACS Contract, Managers have learned to look at each new requirement as an opportunity for economies. When possible, the process is reviewed and changed to accommodate the work. Adding costs without real necessity is not acceptable. This has resulted in such efficiencies as organization-flattening, job combinations, use of existing systems to eliminate new equipment costs, shift scheduling changes, and overtime elimination. Naturally, this philosophy does not maximize "Sales". However, due to the nature of the Fee structure, it can maximize profits.

4.2. Point 2. Adopt the New Philosophy. ("We need a new religion in which poor workmanship and sullen service are unacceptable".) For Operations, this means finding the causes of these problems and rooting them out. (As will be seen later, Deming believes most of them are with the System, not the people.) It is incumbent on Operations Management to create a fertile place for

innovation and quality, and which is equally hostile to poor attitudes. This requires attention to the special environment in which our people work, to their hours, their training, their level of support, etc.

Both JPL and OAO supervisors in the Division have received special training in the tools, techniques and principles of TQM, from a Consultant secured for that purpose. This top level training has since been augmented by selecting obviously-committed members of the Operations Teams for training, with the same Consultant. As a step toward integrating the program into our "culture", OAO has hired a full-time TQM Facilitator/Trainer to train the remainder of the workforce, conduct facilitated process improvement projects and maintain high program visibility.

4.3. Point 3. Cease Dependence on Mass Inspection. (Quality comes from improvement, not inspection). At first blush, this does not appear to have much to do with Operations. However, if we think about how we perform operations, it comes more into focus. Typically, organizations try for high quality by rework, not prevention. An example might be resorting to replay requests to fill "gaps" in data received from a remote station. This is trying to "inspect-out" errors, rather than build in quality. A more effective method is to determine and correct the cause of the gap. Once we begin to look closely at methods, we will find that there are many opportunities to implement improvement, and eliminate unnecessary QC steps.

4.4. Point 4. End the Practice of Awarding Business on Price Tag Alone. (Instead, seek the best quality in a long-term relationship with a single supplier.) The first step in this process has been taken. Operations "vendors" are generally chosen by competitive process which gives high priority to understanding of the requirements and demonstrated experience, not just cost alone. Also, the resulting contracts are for a fairly long term, some five to ten years. As a next step, I suggest that Operations contracts in general could benefit from the broader use of Award Fee structures to encourage Continuous Improvement, as in the case of the MACS Contract. As indicated earlier, a significant portion of total profit available is determined by proven Productivity Improvement and Quality Enhancement initiatives.

4.5. Point 5. Improve constantly and forever the system of Service. (Improvement is not a one-time effort.). Deming and all other serious TQM

practitioners are committed to the Continuous aspect of Continuous Improvement. Another proponent, Thomas Berry states this principle as: "Quality is A Journey, not a Destination" (Ref 2.) In Spacecraft Operations, this means a commitment to long-term, gradual, sometimes agonizing, change. Successes, sometimes small, will build on one another as the system continuously improves.

Continuous Improvement cannot be maintained, or even proven, without Measurements. One of the toughest hurdles in accomplishing TQM in Operations has been the proper selection of Metrics. Finding the right Metrics involves bringing the people doing the work into the process. Then, the products and customers must be identified, and some recurring activity, such as data item delivery, or command sent, or blocks throughput, chosen for monitoring over time. Next, a baseline must be established. This requires a significant amount of data-taking before any conclusions can be reached. Once the data have appeared to settle out, the system can be said to be "in control", meaning that the excursions of individual points from the statistical Mean are reasonably predictable. At this point, performance of the Metrics should be visually and prominently displayed so that unexplainable excursions will be immediately obvious, and can trigger corrective action.

After a great deal of investigating and baselining, we have been able to identify a single Metric for each Task area, allowing us to track effectiveness and detect anomalies quickly. These are generally related to turnaround times or data throughput.

4.6. Point 6. Institute Training. (People can't do their jobs well unless someone shows them how.) This means planned and structured training. Usually, when a tool or capability is being delivered to Operations, if a cost or schedule overrun occurs the first thing to be cut is Training. (The second thing to go is Documentation). The Operators are then left to learn on the fly, or not at all, the best way to operate. Operations Management must have a coherent training plan and must be empowered to pursue it through good and bad budgetary times. Associated with Training is Certification. The customer has a right to expect that critical systems and events will be controlled only by people who have demonstrated their competence.

4.7. Point 7. Institute Leadership. (Leading consists of helping people to do a better job). One of the most valuable principles of TQM is the new Leadership model. Too often, we fall back on the traditions of the Military model of a distant, fault-finding authority figure. This may be the description of the boss we all grew up with, but it is flawed in serious ways. (For one thing, the workforce of the Nineties are far less patient with autocratic Leaders than most of us were.) It is now a well-established fact that collaboration and empathy are superior methods of leading any group, especially one with such a critical mission as ours.

In our local Mission Operations activity, both JPL and OAO Managers have worked to move toward a more mentor-oriented style. For example, we form multi-level Process Improvement teams, in which supervisors and operators serve as peers. This creates a sense of familiarity and empowerment on the part of the people on the line. We also promote from within whenever practical, so that in many cases our first-line Supervisors have recently been operators, and can identify directly with the working level problems. These and other such initiatives have paid off in better communications and more service.

4.8. Point 8. Drive Out Fear. (People need to feel secure to ask "dumb questions" and learn from mistakes). During the early days of Spacecraft Operations, a senior executive in a former company had a standard response when informed of any error in operations, i.e. "Who are you going to fire?" We waste a valuable resource when we cultivate a fear of being wrong. The Total Quality paradigm says that we should not bury our mistakes but learn as much as possible from them. We should encourage the most Junior console operator to ask the most Senior Analyst, "why are you doing that?"

4.9. Point 9. Break Down Organizational Barriers. (One Department's goals may cause trouble for another.) This could have been written with Space Operations in mind. All of us have lived through horror stories about barriers. One of the most recurring is the case in which Development builds something which Ops can't use, but doesn't find out about until far too late. How many of you have seen the six-stage cartoon of building a tree-swing? We need to find ways to see that Operations people, at the Console-operator level, are a genuine part of the Requirements Definition, Specification, Testing, and Acceptance of their vital tools.

Another area in which barriers need to be broken down has to do with inter-organizational rivalry. Progress in barrier breaking can often be made by grouping related Operational activities at the lowest possible organizational level. As layers increase, the time and hassle required to react increase exponentially.

At JPL, we are making strides toward involving Operators in planning for new systems, but accountability to Operations is still lacking. In addition, much more needs to be done to optimize organizations for effective operations.

4.10. Point 10. Eliminate Slogans, Exhortations, and Targets for the Workforce. (Let workers formulate their own slogans.) The greatest dangers to a permanent quality program are sloganeering and faddism. Most of us have lived through "zero-defects", "Value Engineering", and a host of other cure-alls, which faded away along with the banners outside the office entrance. Once our people sense our trust, they won't need slogans to perform. It will be intrinsic. We have found on our contract that an ounce of involvement is worth a ton of slogans. We have seen a high level of commitment to quality principles simply as a result of presenting people with opportunities for training, improvement projects, and general participation in the TQM program. In other words, Empowerment.

4.11. Point 11. Eliminate Numerical Quotas. (They are usually a guarantee of inefficiency and high cost.) This point has relevance to much of the Space Operations environment. Imagine that a Programmer is rated solely on the number of Lines of Code she produces in a day; or that an analyst is penalized at raise time if he delays transmitting more than one command out of a hundred sent. These quotas will totally obscure the systematic realities behind the effort, and be a formidable block to efficiency and perhaps even Mission Safety. A valuable adjunct to this point is Deming's "85-15 Rule". This says that 85% of what goes wrong is due to the system, and is totally outside the human component. It is not only unfair but morale-defeating to measure people against events and situations beyond their control. We should train ourselves to look for the Systematic problems first. It's simple statistics.

4.12. Point 12. Remove Barriers to Pride of Workmanship. (People are eager to do a good job and distressed when they cannot.) Deming stresses the importance of thinking through our policies, and our practices. Providing Operators with poor supervision, faulty systems, and poor visibility into overall program goals are examples of such barriers. Another TQM term which applies here as well: "Recognize and Celebrate". Nothing has more leverage than a well-placed recognition ceremony or bonus check by which a quality performer can be recognized among his and their peers.

On our JPL contract, OAO has moved partially away from Performance ranking through monetary recognition of teamwork and quality program support. We still have a distance to go, however. Questions still exist about how to compensate "teams" instead of individuals. We have seen positive results from our Award Fee sharing program, Employee of the Month and Year, and our annual performance bonus. Through these and other recognition methods, we have been able to leverage a relatively small amount of money into a highly visible and effective program for improvement.

4.13. Point 13. Institute a Vigorous Program of Education and Retraining. (Both management and the workforce will have to be educated.) Total Quality may be only common sense, but it is frequently not intuitive. In order to reach any real level of improvement, both learning and "unlearning" must occur over a broad front. Here at JPL, as well as other centers, a great deal of commendable effort has already been expended in retraining staff at all levels to the new Quality paradigm. In Operations, we must make special efforts to move key people off the consoles and into classrooms, then follow up with practical training at their workplace. We have found that Process Improvement projects, initially under the guidance of specially-trained Facilitators, is an excellent way of validating the classroom training.

On our JPL contract, OAO has appointed Quality Subcommittees in each major line organization. In Mission Operations, this subcommittee has been trained, and is now involved in defining candidate processes for improvement projects. It is convened at least monthly by the Manager for general discussion and work on activities under way.

4.14. Point 14. Take Action to Accomplish the Transformation. (A critical mass of people must support the goal.) This point really summarizes the necessity of doing the other thirteen points right. One truth has been found wherever a Total Quality Program has been tried, e.g.: It takes more than a few “missionaries” to make TQM work. While a few may catch the vision first, there is no point in trying to achieve any real results without “selling” the program to the workforce at large. The good news is that the process is contagious. Even cynics begin to see the benefits of the program once it is underway. It is also a proven fact that TQM will not work without top-level support. Unless the people with real power are behind it, it will fail.

In large measure, the success of the MACS contract’s Program has been due to the active leadership of senior JPL and OAO management who support its goals with their continued attention, tough questions, and the necessary financial and human resources.

5. RESULTS ON THE JPL/OAO CONTRACT

Since assuming responsibility for the MACS Contract, OAO has submitted and had approved for Award Fee consideration, over 100 Process Improvement initiatives. These have resulted in cost savings to the Government of over \$9 Million , to which can be added numerous examples of improved processes and enhanced mission reliability. As a result , OAO has earned an additional \$1.3 Million in Profit, a significant percentage of which we have shared with our employees. And through this unique partnership, we have created an environment in which all parties have a reason to share in the bright future offered by Total Quality Management.

6. CONCLUSION

Based on personal experience gained in JPL Mission Operations, as well as the documented success stories wherever it has been tried, I am convinced that TQM can be a valuable tool for Spacecraft Operations. It can provide us with a valuable vantage point from which to achieve greater effectiveness, and thus greatly assist in reversing the operations cost trends without introducing additional mission risk.

7. REFERENCES

1. Walton, Mary. 1990. Deming Management at Work. Putnam’s & Sons
2. Berry, Thomas. 1991. Managing the Total Quality Transformation. McGraw/Hill

MISSION OPERATIONS AND COMMAND ASSURANCE: INSTILLING QUALITY INTO FLIGHT OPERATIONS

Linda L. Welz, Mona M. Witkowski, Kristin J. Bruno, Sherrill S. Potts

Jet Propulsion Laboratory
California Institute of Technology
4800 Oak Grove Dr., Mail Stop 125-233
Pasadena, CA 91109

Abstract

Mission Operations and Command Assurance (MO&CA) is a Total Quality Management (TQM) task on JPL projects to instill quality in flight mission operations. From a system engineering view, MO&CA facilitates communication and problem-solving among flight teams and provides continuous process improvement to reduce the probability of radiating incorrect commands to a spacecraft. The MO&CA task has evolved from participating as a member of the spacecraft team, to an independent team reporting directly to flight project management and providing system level assurance. JPL flight projects have benefited significantly from MO&CA's effort to contain risk and prevent rather than rework errors. MO&CA's ability to provide direct transfer of knowledge allows new projects to benefit from previous and ongoing flight experience.

Key Words: Mission operations, command assurance, Total Quality Management, defect prevention, error management, system engineering

1. Introduction

A long-term program is in progress at JPL to reduce cost and risk of flight mission operations through defect prevention and error management. Flight mission operations require systems that place human operators in a demanding, high risk environment. This applies not only to mission controllers working in the "dark room" and Deep Space Network (DSN) operators configuring and monitoring DSN operations, but also to flight teams that plan the mission and develop the command sequences and to engineering teams responsible for analyzing spacecraft performance. The flight operations environment generally requires operators to make rapid, critical decisions and solve problems based on limited information, while closely following standard procedures (Refs. 1-3). The mission operations environment is, therefore, inherently risky because each decision made is potentially mission critical.

To contain this risk at JPL, flight mission operations procedures (as described in Refs. 4-5) currently require intensive human reviews. In addition, when an error does occur, rapid rework is required to ensure mission success. This strategy has worked well to reduce risk and has ensured the success of JPL missions. However, the large human labor investment required for review and rework has substantially contributed to the overall cost of flight mission operations. Prevention of errors would greatly reduce both cost and risk of flight projects. The motivation of the long-term defect prevention/error management program is to contain risk in a more cost effective manner by preventing errors rather than reworking them. The goal of this program is the management, reduction and prevention of errors.

A major element of this program is the Mission Operations and Command Assurance (MO&CA) function. MO&CA provides a system level function on flight projects to instill quality in flight mission operations. MO&CA's primary goal is to help improve the operational reliability of projects during flight. MO&CA's effort focuses directly on continuous process improvement to reduce the probability of radiating incorrect commands to a spacecraft. MO&CA occupies a unique position in the flight project organization, reporting to both flight project management and the System Assurance Division of the JPL Office of Engineering and Review. As a result, MO&CA is able to cross operational boundaries between teams and offices on a flight project. This system level view enables MO&CA to enhance inter-team communication to facilitate problem solving within the project.

This paper describes the development and evolution of the MO&CA function at JPL and the benefits provided to flight projects by MO&CA.

2. Evolution of the MO&CA Task

The MO&CA task began on the Voyager (VGR) project in 1985. In response to an increase in command related problems a study was conducted by the JPL Office of Engineering and Review to review

the adequacy of procedures, operations and software involved in real-time commanding with the goal of reducing errors. Incident Surprise Anomaly (ISA) reports, problem reports written by flight team members when an anomaly occurs in flight operations, were analyzed from a seven year period (1978 - 1985) to determine the causes of command errors. Based on this analysis, recommendations were made to the project for improvements in flight procedures including: upgrading the command development software to improve readability of command printout thus facilitating command reviews and approvals; providing traceability between command forms and ISAs to facilitate analysis and correction of command incidents; reducing real-time commanding by improving the coordination of real-time and sequence commanding and including as many commands as possible in command sequences; and updating flight team training to include command awareness issues making flight team members aware of potential command problems and how to avoid them. Command development procedures were updated to incorporate these recommendations.

When an opening occurred in the spacecraft team the following year, the position was filled by a MO&CA engineer who became the Systems Lead for real-time commanding for the VGR Project. MO&CA also continued to analyze ISA reports and make recommendations for continuous improvement to the commanding process.

MO&CA provided both a system engineering function for the spacecraft team and a systems assurance function for the VGR Project. Although not included in the project at launch, the MO&CA function was successfully integrated into the spacecraft team. This placement allowed MO&CA to not only recommend changes to command procedures, but also to implement these changes, with project management concurrence, for the spacecraft team.

Following VGR, a MO&CA team was activated on the Magellan (MGN) Project just two months prior to launch, on May 4, 1989. The main MO&CA task for MGN was to detect, analyze and correct defects that existed in flight operations and procedures. Tasks to detect and analyze defects included: review of flight operations processes and documentation; participation in operational readiness training exercises; assessment of command process compliance with requirements and procedures; analysis of ISA reports; and participation in software testing. Activities associated with defect correction included: command awareness training; real-time support to the command development process; participation in the analysis and correction of

command incidents; implementation of an Anomaly Reporting and Tracking System (ARTS); and support for investigation and development of workarounds for hardware failures.

One of the major efforts of the MGN MO&CA team was assisting the flight project to upgrade the real-time command process and related operational procedures. The initial real-time command process in place at launch included only a handful of steps. Commands were initiated within one subsystem of the spacecraft team, passed to another subsystem for coding, and submitted to project management for approval to transmit. No system level effort for review or coordination was in place.

In the first few months following launch the real-time command system was in constant use due to several hardware problems on the spacecraft. Extensive operational workarounds were required to compensate for these problems. Because of the anomalies and a lack of coordination in the real-time command process, command problems occurred. MO&CA recommended improvement to the command process which included: review of commands by all subsystems prior to development; system level coordination of all commanding; management approval prior to command development; traceability of commands from initial request to final approval for transmission; development of rigid test requirements for all commands; required representation by all flight teams at command review and approval meetings; spacecraft team support of the command coordination meeting; training for all flight team members with the newly developed command procedures.

By December, 1989, an updated real-time command process was in place on MGN. This well-structured process enabled the flight team to function well as a unit and respond quickly to anomalies. Real-time command incidents decreased dramatically despite the fact that the flight team continued to face spacecraft anomalies.

In contrast to the VGR project, the MGN team was not integrated directly into an existing team on the flight project, but instead formed an independent unit. While this enabled the MO&CA team to maintain a systems view of flight operations, it did not provide the same ability to implement changes. MO&CA instead provided recommended changes based on ISA analysis and direct participation in working, review and approval meetings. The flight team, directed by project management, implemented the changes to operations procedures and processes.

Due to the success of the first MO&CA teams, the MO&CA function was proposed to the management of the TOPEX/POSEIDON Project. The concept was well received by the Project and in November 1991, a MO&CA engineer was placed on staff to the TOPEX/POSEIDON Flight Operations System Manager to facilitate development of the Real-time Command Process.

A unique aspect of the TOPEX/POSEIDON Project is the amount of planned real-time command activity. Due to the nature of the mission, TOPEX/POSEIDON is very dependent on real-time command activity. For example, tape recorders are played back via real-time commands three times per day. Once in place on the project, MO&CA quickly assessed existing flight operations plans and noted that an additional process for the development and approval of unplanned real-time commands was required.

MO&CA worked with the flight team to define required interfaces for the unplanned real-time command process and develop the necessary procedures and process descriptions. While the flight teams were preparing individual team operating procedures, MO&CA was able to provide a system level overview and develop the additional process and procedures that cross team and division boundaries. As the project planned to use the real-time command process extensively, MO&CA coordinated the development of a Real-time Command Library. This library consisted of all pre-defined real-time command files that were planned for repeated utilization throughout the life of the mission.

In addition to the Real-time Command Process and Library, MO&CA also developed the Operations Coordination Process, Anomaly Reporting Process and aided in development in the Configuration Management Process. As a portion of the Operations Coordination Process, MO&CA documented the types and level of necessary Flight Operations meetings.

By January 1992, the TOPEX/POSEIDON Real-time Command Process and Library were in place. The transition to test and training activities went smoothly. The flight team readily adapted to, and exercised the Real-time Command Process, the Real-time Command Library, and the Anomaly Reporting Process. With a functioning pre-launch Anomaly Reporting Process for ISA reporting, MO&CA was able to gather and present detailed statistics to the project on recurring problem areas encountered during test and training.

Since launch on August 10, 1992, MO&CA's efforts have been focused on coordination of the real-time command process, coordinating approval and uplink. MO&CA also plays a key role in sequence review and validation of all planned real-time commands.

The most beneficial portion of the Real-time Command Library proved to be the Contingency Commands. The Contingency Commands were invaluable when spacecraft anomalies occurred early in the mission, facilitating recovery operations during a high activity period. The value added by the MO&CA Real-time Command Library is also visible daily during mission operations. The majority of planned real-time commands in the TOPEX/POSEIDON Sequence of Events are pulled "off-the-shelf" from MO&CA's Real-time Command Library.

The TOPEX/POSEIDON Project experienced an immediate benefit through the direct transfer of MO&CA's knowledge and experience from previous projects. The MO&CA task combined elements from the VGR and MGN MO&CA experience. Like the MGN MO&CA team, TOPEX/POSEIDON MO&CA functions as an independent unit, and, like VGR MO&CA, TOPEX/POSEIDON MO&CA has the ability to implement improvements in flight operations procedures. TOPEX/POSEIDON MO&CA was the first MO&CA team to be in place on the flight project an extended time prior to launch. The team was therefore able to implement "lessons learned" and process improvements early while flight operations procedures were being developed. This opportunity allowed MO&CA to instill quality into the flight procedures in a pro-active manner, rather than work reactively to update processes and procedures after completion of mission operations development.

3. Benefits of MO&CA

The MO&CA function has evolved from participating as a member of one flight team on the VGR project; to an independent unit providing assessments and recommendations based on a systems view to the MGN project; and finally to its current state as a combination of both direct flight team participation and system level assurance on the TOPEX/POSEIDON project. MO&CA's unique position, with both a high level systems view and the direct ability to develop and implement changes as necessary, enables MO&CA to provide flight projects the benefits of both the assurance and system engineering functions.

Originating from the Systems Assurance Division at JPL, MO&CA provides a direct transfer of

knowledge between current missions in addition to providing valuable "lessons learned" experience to new flight projects. New projects are able to thus benefit directly from both previous and ongoing mission operation experience. Lessons learned can be incorporated early on, into project requirements, thereby eliminating the amount of necessary rework on flight operations procedures. The real-time command process and library on the TOPEX/POSEIDON project are examples of this direct transfer of knowledge.

Process improvement activities require the ability to measure and evaluate a process. MO&CA teams collect and analyze error data from the ISA reports written by flight teams on operational problems. Many of MO&CA recommendations for process improvement are based on these reports. This error analysis results in improvements not only to the project that wrote the report, but also to other flight projects via transfer of knowledge. The error analysis information is also used for analysis in the overall defect prevention / error management program.

MO&CA's unique position as an independent unit in the flight project organization provides its ability to facilitate communication and problem solving. Problems that span many teams and offices within a flight project can be effectively addressed by MO&CA. Coordinating real-time command processes is an example of this task. Flight project members who are faced with problems that impact several teams, often bring the issue directly to the MO&CA engineer when they cannot be addressed solely by their team. MO&CA is also able to improve the efficiency of data reporting that crosses team boundaries. The newest MO&CA team on the

Mars Observer (MO) project noted that identical data was being reported by several of the flight teams, and that the data was often erroneous if it originated in one team and was reported by another. MO MO&CA worked with the teams to eliminate this duplication and ensure correct data was reported.

4. MO&CA and TQM

The MO&CA function is one example of a Total Quality Management (TQM) process at JPL. Specifically, MO&CA embodies the TQM principle of continuous process improvement (CPI) in which processes are continuously examined and analyzed for opportunities for improvement. Figure 1 shows how MO&CA implements CPI in two ways. First, within ongoing projects, the flight mission operations environment is established and MO&CA participates as a team member. In the course of day-to-day operations, anomalies are documented as ISA reports. The ISAs then serve as data which is analyzed by MO&CA engineers for process improvement opportunities. When these opportunities are identified, MO&CA provides reports and data to support recommendations for improvement to project management. Finally, based on management approval, MO&CA helps the project implement the changes back into the day-to-day mission operations environment. This technique was successfully implemented on the VGR, MGN and TOPEX/POSEIDON projects.

The second way in which MO&CA implements CPI on JPL projects is on new projects or upgrades to existing projects. The recommendations that are

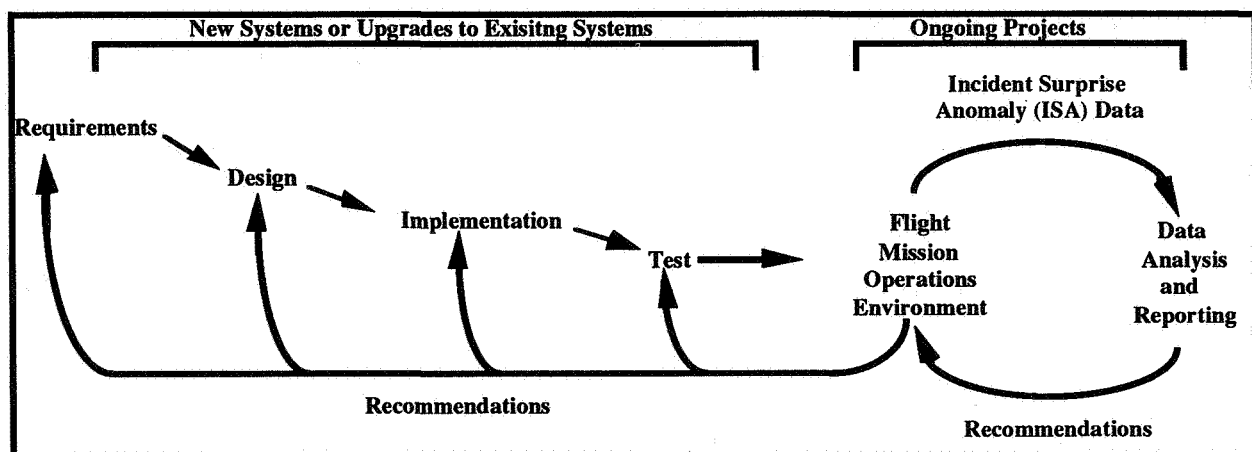


Figure 1
TQM Model of MO&CA

developed from the data analysis on ongoing projects are used as input to system requirements on new projects. This allows new projects to benefit from improvements made on past projects as TOPEX/POSEIDON benefited from the experience gained on VGR and MGN.

Using this technique, not only do ongoing projects continuously improve, but each new project starts with a better set of requirements and better processes than the last one. At JPL this continuous improvement feedback loop has improved flight mission operations processes from the Voyager Project, to the Magellan Project, and to the TOPEX/POSEIDON project. Additionally, this continuous process improvement reduces cost and risk of flight mission operations.

5. The Future of MO&CA

Future flight missions at JPL will have smaller spacecraft and flight teams (Refs. 6-7). Development times will be reduced and the teams that design and build the spacecraft will also staff the flight mission teams. MO&CA will need to evolve to adapt to this changing flight operations environment

MO&CA will continue to provide both system assurance and engineering assistance to operations. The MO&CA function will be implemented during project development so that MO&CA can assist in developing operational procedures and participate in flight team training. This will streamline procedure development and eliminate late changes and upgrades thus reducing rework and cost.

Automation of data tracking and analysis by MO&CA help to make operations process monitoring and error analysis more efficient and timely. With automation, MO&CA will be able to address problem areas quickly. Finally, ISA data will be used in a parallel error analysis study. The findings of this study (Ref. 8) will enable prevention of errors through improved requirements development on new projects.

6. Summary

JPL flight projects have benefited significantly from MO&CA's effort to contain risk and prevent rather than rework errors. MO&CA's ability to provide direct transfer of knowledge allows new projects to benefit from previous and ongoing flight experience. The system level view of project operations provided by MO&CA enhances communication to facilitate problem solving within a flight project.

MO&CA will continue to evolve to meet flight project needs. Early involvement with developing projects will ensure that quality is incorporated into mission operations during operations development and training.

The MO&CA function at JPL has built quality into mission operations, enabling flight teams to operate efficiently and effectively in a dynamic space flight operations environment. Since MO&CA, as a TQM effort, focuses on continuous improvement of processes and elimination of rework, MO&CA will continue to provide benefits to flight projects.

7. Acknowledgment

The work described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

The effort of many people from the JPL Office of Engineer and Review went into the MO&CA work on JPL projects. The authors would like to acknowledge their efforts by recognizing the participants: Al Brejcha, Anna Bruhn, Grant Faris, Larkin Hamilton, Sara Hyman, Kil-Sun Kang, Farinaz Kavousirad, Sheri Kazz, Young Kim, Robyn Lutz, George Nichols, Irwin Plitt, Anne Phu, Richard Santiago, Mitch Scaff, John Schlue, Hui-Yin Shaw, William Shipley, Lowell Thompson.

8. References

1. Horvath, J.C., and Perry, L.P. 1990. Hypercubes for critical spacecraft verification. In *Proceeding of AIAA Second International Symposium on Space Information Systems*, Pasadena, Ca., 1285-1291.
2. Linick, T.D. 1985. Spacecraft commanding for unmanned planetary missions: the uplink process. *Journal of the British Interplanetary Society*, 38:450-457.
3. Muratore, J.F., Heindel, T.A., Murphy, T.B. Rassmussen, A.N. and McFarland, R.Z. 1990. Real-time data acquisition at mission control. *Communications of the ACM*, 33(12):18-31.
4. Haynes, N.R. 1985. Planetary mission operations: an overview. *Journal of the British Interplanetary Society*, 38:435-438.
5. McLaughlin, W. 1987. How to feed a spacecraft. *Spaceflight*, 29: 38-40.

6. Cassini, K., and Spear, A.J. 1992. Low Cost Spacecraft: The Wave of the Future [Videotape]. Pasadena, Ca.: Jet Propulsion Laboratory.
7. Stone, E.C. 1992. Total Quality Management Seminar [Videotape]. Pasadena, Ca.: Jet Propulsion Laboratory.
8. Bruno, K.J., Welz, L.L., Barnes, G. M., and Sherif, J.S. 1992. Analyzing human errors in flight mission operations. A Paper presented at the Sixth Annual Space Operations, Applications, and Research Conference, NASA Johnson Space Flight Center, Houston, Texas, August 4 - 6.

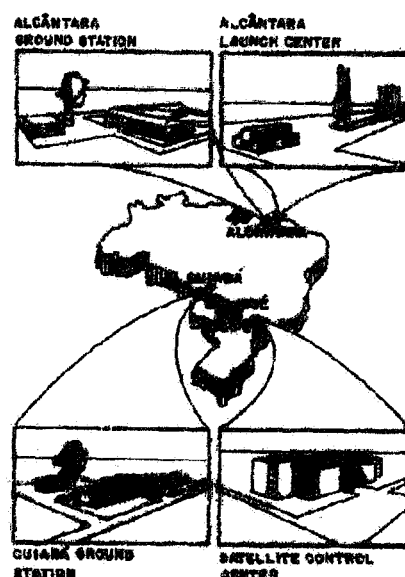
N 9 4 - 2 3 9 3 3

OPERATIONAL TRAINING FOR THE MISSION OPERATIONS AT THE BRAZILIAN NATIONAL INSTITUTE FOR SPACE RESEARCH (INPE)

Pawel Rozenfeld
Brazilian Institute for Space Research
São José dos Campos, SP, Brazil

ABSTRACT

The paper describes the selection and training process of satellite controllers and data network operators performed at INPE's Satellite Tracking and Control Center in order to prepare them for the mission operations of the INPE's first (SCD1) satellite. An overview of the Ground Control System and SCD1 architecture and mission is given. Different training phases are described, taking into account that the applicants had no previous knowledge of space operations requiring, therefore, a training which started from the basics.



1.0 INTRODUCTION

In order to control its satellites in orbit, the Brazilian National Institute for Space Research (INPE) had established a ground control system called Satellite Tracking and Control Center (CRC). It is composed of a Satellite Control Center (CCS) located in São José dos Campos (23 17 S; 45 51'W), a Ground Station located in Cuiabá (15 33'S; 56 04'W) and another Ground Station in Alcântara (02 20'S; 44 24'W) (Fig.1).

Fig.1 - Geographic Distribution of CRC.

Connecting all these sites there is a private data communication network, called RECDAS, and a voice communication system. CRC had no previous experience in controlling satellites in orbit and the launch of the first (SCD1) data collecting satellite constitutes the maiden operation for CRC. This paper describes the selection and training process of spacecraft controller (CONSQT) and RECDAS and computer operators undertaken by CRC for this mission. It should be kept in mind that there are no experienced people for these positions readily available in Brazil and the training started from the very basics.

Also, the late delivery by the software developers of the satellite simulator and even satellite control software modules made the task of training the operational people quite a difficult one.

A brief idea on the ground control system and the SCD1 main subsystems and the Satellite mission is given in order to define better the boundary conditions for the training.

2.0 GROUND CONTROL SYSTEM OVERVIEW

In this section we define the scheduling problem beginning with a simplified version and evolving to a more realistic definition.

The Ground Control System was conceived to keep high its availability and at the same time, to keep to a minimum the man-power required to operate it. This premise reflects itself in the overall architecture of the system: there are two redundant Ground Stations and redundant equipment at the Satellite Control Center (CCS) as well as two physical lines in the RECDAS.

The Satellite Control Center hardware is composed of a DEC VAX11/780 and two VAX8350 computers. The first one is dedicated to software development while the other two are dedicated to software development while the other two are dedicated to mission operations. In order to increase system availability, two VAX8350 are in cluster arrangements. There is also volume shadowing software implemented increasing in this way disk availability. The operational system used is VMS 4.7 version.

The CCS application software is a brand new one developed in house. It is written in FORTRAN. The communication between CCS software and Ground Station equipment is done according to ESA SDID protocol.

Going now to the Ground Station hardware, its front end and receivers are manufactured by Scientific Atlanta (USA). The High Power Amplifier is manufactured by MCL (USA). The telemetry baseband equipment is a mixture of LORAL 9USA), Inter technique (France) and in-house built equipment.

In order to minimize the operational man-power requirements, the Ground Stations hardware is configured and monitored by DEC VAX II computer which can also assume some of CCS functions in emergency situations.

The Telecommand, Telemetry and Ranging functions are performed according to ESA PCM standards.

Insofar as the data communication is concerned, it is a private packet switching network implementing X.25 access protocol with bit rate of 9600 bps between the nodes. The nodes are built by a Brazilian manufacturer compatible with French SESA nodes. The network management computer is COBRA, a Brazilian-built computer.

3.0 THE FIRST (SCD1) SATELLITE AND ITS MISSION

The mission of SCD1 satellite is environmental data collecting. A network of automatic Data Collecting Platform (DCP) is scattered throughout Brazilian territory. They collect and transmit to the satellite in UHF band environmental data (temperature, pressure, wind speed, river level and so on).

SCD1 carries as its payload a frequency transponder which transposes the collected data to the S-band. When the DCPs, SCD1 and the Ground Station are in mutual visibility the data are received by the Ground Station which transmits them, via RECDAS, to the Data Processing and Distribution Facility. The last one processes and distributes the data to the users.

SCD1 is a low earth orbiter. Its altitude is 750 km and its inclination is 25°. In this way whole Brazilian territory is covered, assuring at least one contact per day with the DCPs.

The satellite has an octagonal prism shape with base diameter of 1m, height of 1,055 m and its weight is 115 kg. Satellite's all but one faces are covered by solar cells. The remaining face is used as a heat sink. It should not be directed to the sun. In order to avoid this problem an attitude manoeuvre is foreseen after 6 months in orbit. No other manoeuvres are planned for the satellite. The thermal control is passive. The attitude subsystem is constituted by a magnetometer and two sun sensors.

The TT&C antennas are located on the opposite bases. Because of their radiation patterns silent zones appear during the passes over the Ground stations. This fact has to be considered by the Ground Control System in planning the daily activities.

The two experimental subsystem on board of the satellite are the on-board computer and the solar cells developed in-house and they are to be flight qualified. The on-board computer is used only to turn on and off

the solar sensors and magnetometer and to perform tries on computer memory loading.

The satellite will be injected in its orbit in such a way that the Alcântara Ground Station will be the first Ground Station to track the satellite. Cuiabá Ground Station will track the satellite starting from the third orbit. There will be no tracking support from foreign Ground Stations. NORAD will provide initial orbit information.

4.0 OPERATION WORKING MANNER AND CANDIDATE SELECTION PROCESS

SCD1 passes over the Ground Stations, as it happens with all low earth satellites, are short ones, in this case, of order of 10 minutes and they drift in time.

There is a visibility gap of approximately 9 hours every day when the satellite is not visible by the Brazilian Ground Stations. This visibility gap could not be used to reduce the number of working shifts because it is drifting every day (Fig. 2). This would require the working shifts to be drifting too. As a consequence it would make difficult for the controllers to organize their personnel life. Therefore, it was decided to run the satellite control on 24h/24h basis in six hours fixed shifts.

For security reasons there should be two operators per shift at the same site.

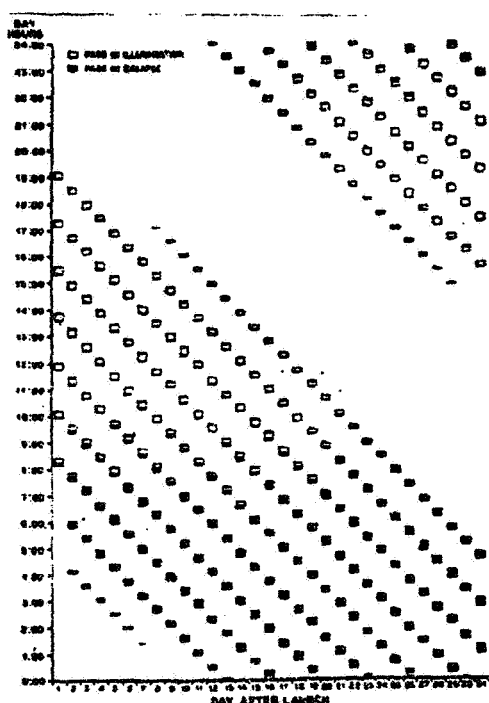


Fig. 2 - SCD1 passes over Cuiabá Ground Station.

Some man-power reduction have been planned taking advantage of CCS building layout. The RECDAS Network Management Center and São José dos Campos communication node are located near the VAX computer room. Due to this fact and the very simple to operate, it was decided that RECDAS operators should do also the job of VAX operators.

The Satellite Control Room is located above the VAX computer room and, therefore, there should be 2 CONSATs per shift.

The operational needs determined the initial educational level of the prospective operators and controllers. The minimum requirement was to have completed high school education with good scholastic standing. In the process, some college students were selected.

As far as the CONSAT position was concerned, it was decided that more software-oriented people should be selected. For the RECDAS operator position, people with a background in hardware (electronic technicians) were to be selected because it required dealing with protocol analysers, modems, data sets, and so on.

The selection was based on the C.V.s, interviews and psychological tests allowing candidates to decide on one of the available working shifts. The psychological tests included auditory and visual memory tests, concentrated attention tests (Toulouse-Pieron), personality tests (R.B. Cattell & H.W. Eher). An essay done by candidates complemented the personality test allowing, also, to perform some graphology analysis. Although there was no preference for male or female candidates, most of the satellite controller positions were filled by female candidates and the network/computer operators by male candidates, reflecting, perhaps, the educational preferences of Brazilian youth at the present time.

5.0 Basic Training

Since none of the selected people had any idea whatsoever about how the satellite looks or works or why it stays in orbit, it was decided to give them initially a general training module on satellites and their missions, satellite architecture, flight dynamics, CRC organization, satellite control team organization, etc. This module took about 45 hours of classroom presentation. After finishing this general module, the training was split into two branches: one for CONSAT and the other for the RECDAS/VAX operators.

The CONSATs had a module dedicated to the operation of the satellite control software. It took 30

hours of classroom explanations, followed by the same number of hours of hands-on training. In this way the CONSATs were acquainted with the satellite control software, booting, screen layouts, CCS computer connection with the Ground Station equipment, telemetry reception and processing, history files archiving and retrieval, telecommand generation and transmission, ranging requesting, alarms, and their meanings, file transfer, etc. During and at the end of this module, an evaluation of the student's knowledge was performed.

Another module was dedicated to the Flight Dynamics software. It comprised 25 hours of theory and 15 hours of practice on such matters as predics generations, orbital and attitude data pre-processing and attitude determination. This was more of a get-acquainted module because this software is actually operated by more specialized people.

The following training module was on SCD1 architecture, its specific telemetries and telecommands and operating modes (launch, stand-by, orbit determination, attitude determination, normal, experimental, experimental standby, manoeuvre and manoeuvre standby). This module took 35 hours of lectures. Subsequently the students used a Portable Simulation System based on a PC to practice on SCD1 telemetries and telecommands.

During this training period, CONSATs attended other activities which had been performed at the CCS at that time. These were the Cuiabá Ground Station calibration using real satellites in orbit (ESA's Hipparcos and ERS1) and SCD1 system test.

The RECDAS/VAX operators started their specialized training with a module on data communication, networks and protocols including X.25. Node hardware and software, NMC computer and software, modems, monitor and test tools and operational procedures were also part of this module. It took 25 hours of theory and 30 hours of practice. Several evaluations of student performance were done. A need for supplementary training on data communication was detected; therefore, a supplementary 8 hour module was offered.

Another module on general computer architecture and specific VAX operation was given. It took 10 hours of classroom lectures and about the same number of hours for hands-on practice. Then, the time came for the RECDAS/VAX operators to start to work on the shifts. For about one month, they worked under supervision of more experienced operators and then they assumed the duties on their own.

All contracted people were trained in the operation of voice system, air conditioning and no-break system of the CCS building.

6.0 Mission Operation Training

The mission operation training was based on the SCD1 software simulator. This simulator was developed in-house. It gives CONSATs a realistic operational environment generating realistic telemetries and reacting in a realistic manner to the telecommands. It simulates possible satellite failures and reproduces, within certain limits, realistic on-board thermal and power generation conditions. They are determined by the SCD1 orbit and attitude which are also simulated. The passes over Ground Stations are simulated. This means that some Ground Station equipment had to be included in the simulation process.

The operational training of the CONSATs started with a review of the satellite control software operation and SCD1 telemetries and telecommands because of the late delivery of the simulator. The CONSATs trained in the nominal satellite operating modes and transition between them. This process allows CONSATs to memorize the subsystem telemetries which define each one of the operating modes and the telecommands which cause transitions between them. After performing a mode transition, a detailed discussion of the effect of each one of the telecommands on the SCD1 subsystems was done, using satellite electrical architecture drawings for system visualization. During this part of the training the satellite simulator was set to a mode in which there was a continuous satellite pass over a Ground Station. This part took about 20 hours of the training.

A second phase in the training started imposing a time constraint (satellite passes) on the CONSAT activities, using different time lengths for the satellite passes. The CONSAT had to perform ranging requests, several modes of telecommanding and telemetry limits checking. The simulator was set to satellite nominal operation. This phase required 15 hours of training.

From that point on, failures in the subsystems were introduced. These were the failures covered by the satellite EMECA. The CONSATs reaction to the failures were observed. Although it was not expected that in real operation the CONSATs by themselves would assume the necessary recovery procedures, it was considered that CONSATs must be exposed to these failures in order to assure their self-confidence during unexpected events in real operations. This phase took 20 hours of training.

Finally, before starting the rehearsals, a training which involved whole Ground control system (Satellite Control Center, RECDAS and Ground Station) was performed. this training made use of the satellite suitcase model located at Cuiabá Ground Station and connected to its front end. The CONSATs performed suitcase model telecommanding, received respective telemetries and requested a group of ranging measurements. As mentioned before, during their basic training the CONSATs attended the Ground Station calibration by means of ESA's Hipparcos and ERS1 satellites as well as SCD1 system tests. This time it was the first time the CONSATs as well as RECDAS operators and Ground Station operators were involved in an end-to-end training. This training was repeated twice.

At this point, the CONSATs were considered apt to be a part of the spacecraft control team and they participated in the mission rehearsals.

7.0 Conclusions

The launch next December of the first satellite developed and manufactured by INPE will constitute a major endeavor for its new Satellite Tracking and Control Center (CRC).

The selection and training of the satellite controllers (CONSATs) and RECDAS/VAX operators, all of them having no previous experience in this kind of work, constituted a major challenge to CRC. It should be mentioned that delayed delivery of the Satellite Control Software and SCD1 simulator added an extra constraint to the training process.

As it appears now the selection process was acceptable because out of 17 people selected initially only one was not able to meet the requirements.

It was observed that CONSATs needed constant review of the satellite control software operation. It is expected that this need will decrease over the long term when people become more mature in the CONSAT function. In addition, it was noted that a good knowledge of the Ground Station hardware by the CONSATs helps to speed up the process of learning the satellite control software.

TRAINING: PLUGGING THE INFORMATION GAPS **N 9 4 - 2 3 9 3 4**

By Carol J. Scott

Jet Propulsion Laboratory, California Institute of Technology
Pasadena, California, USA

ABSTRACT

Training is commonly viewed as an add-on function to the development cycle. It is imperative that this view be changed. The training developer needs to be a colleague in the system development process, contributing and learning along with the other development participants.

Training developers can make contributions to design concepts that favor end users. Early involvement will enhance the likelihood of training availability concurrent with software delivery. End users will benefit and cost-savings will be realized.

1. INTRODUCTION

As software training engineers responsible for workstation course design and implementation, we have an obligation to provide our customers with information and learning experiences that translate directly to their working environment. "Customer" is the key word, and, at the Jet Propulsion Laboratory, our customers are the flight projects. They require a supportive, well-informed, educational resource.

Training will be a more effective resource when we are active participants in project development activities. Without interaction in system development, the thought processes that justify the design are lost to us and we have a lack of perspective on resulting software products. Many design and interface decisions are made without our representation, decisions that will directly affect training quality. Training engineers are overlooked in the early phases of software development because of the lack of a "visible need" for our attendance at the design table. Funding sources look for immediate evidence of their working dollars, and anticipated productivity for training is low during the development phase.

Training may be one of the last steps on the "development ladder," but it is the first step, and often the most important, for end users.

2. THE TRADITIONAL SOFTWARE DEVELOPMENT CYCLE

The traditional software development cycle begins with a customer's need, formally specified by system engineering in a Functional Requirements Document (FRD). The FRD is delivered to software specialists, who interpret the requirements and make them a functional reality in the form of a software program.

The completed software is delivered for testing, where it must conform to the requirements of the FRD. Testers flag performance failures and the software is "fixed" and redelivered by the Software Developers. The "accepted" version is delivered to the customer, where system administration personnel perform software installation procedures for operational end users.¹

Training developers, who at this point are end users, jump into action to learn how to operate the new software. We also incorporate input from operations personnel on end user tasks to analyze and understand how the software will be used on the job.² Only then can we develop appropriate learning modules and training materials. Training development often exists as an isolated effort which interfaces to the remnants of a development task force who are no longer interested in rehashing the "why's" of its design.

3. DISCOVER THE INFORMATION GAPS

The traditional positioning of training development at the end of the software development cycle limits our level of understanding to whatever comes "out" of the process. Imagine trying to train a quarterback if everything you knew about football came from sports

¹ Delivered software is controlled by Configuration Management; operational procedures are documented and released as user guides.

² New and modified functional requirements will be generated by Operations as user needs change or grow.

recaps on the nightly news. You've got to be in the "locker room" to understand the plays!

3.1 The Customer/Training Information Gap

Space flight projects depend on the Training function to bring their operations and engineering personnel to a state of readiness.

3.1.1 Problems

Training is a multi-mission service provided to project operations and assumed to be available for scheduling when needed. Unfortunately, training engineers often take delivery of the software at the same time the project does and the training development effort may be just beginning. The project hardware and personnel buildup may be nearing completion so we are under pressure to commit to a training delivery schedule.

We frequently know very little about our prospective students. We often "train" new project recruits whose jobs are still undefined; sometimes we get entire teams that won't be receiving computers for months; and occasionally people are sent for training to take every class we offer because they "might need that information someday." Workstation training is an intuitive process that offers a multitude of flexible programs with capabilities and options for handling nearly any given task. The "right way" to perform a task is a subjective decision based on the user's processing goals, an existing set of variables, desired results, and user experience.

Some students may know little or nothing about their jobs. These people come to training expecting us to give them a purpose. Without a thorough understanding of mission and task goals, it is difficult for trainers to recommend a course of action in the learning environment. This comes home to us very clearly when our students pose logical "what, why, when, and how" questions that we are unable to answer.

We don't always know the environmental conditions under which user groups may be operating. Software access methods may differ from one user to the next, even on the same workstation, depending upon each user's environmental configuration characteristics. Individual teams, or positions within a team, may use a common set of programs, but have a unique perspective on the application and a different set of goals.

3.1.2 Solutions

The customer must provide training engineers with a current set of training requirements (or the FRD and an interpreter) to create a bridge between the functional requirements and the positions to be trained. This bridge will present a clearer picture of why our students are at the training table. Updates to these requirements are also needed to keep the training engineers up-to-date.

The customer is the only source of detailed information about what end users need to learn. A Position Description Document (PDD) cross-referenced to the system's functional requirements would be extremely helpful. With this information we could determine specific areas of responsibility and better estimate the level of training required. We could also provide an "educated guess" at the costs of training to that level.

Even with a PDD, a task analysis is difficult to perform for custom software users because of the enormous number of decision points to be considered. One suggestion, to make it easier, would be to use a Software Capability Checklist as a training objective selection tool for interpreting the Position Description Document. This checklist identifies and eliminates potential training objectives which are not addressed by the PDD, then reveals paths to decision points where major options can be associated with specific objectives.

The customer should identify the experience levels of prospective trainees. Classes are more productive when participants have similar knowledge and experience. A rookie is often less willing to ask questions when paired with an expert; likewise, it's difficult to maintain the interest of an expert who is pacing a rookie.

As training engineers, we need a permanent, meaningful point of contact on each project to supply us with information and make decisions for all their users.

3.2 The System Engineering/Training Information Gap

In addition to developing system functional requirements, System Engineers design operational hardware and personnel interfaces.

3.2.1 Problem

There is often a significant difference between the proposed operational working environment and what users can actually access. When there is no

consistency between the learning environment and the working environment learners have no way to apply their new skills. In these cases training can be a costly, frustrating experience.

3.2.2 Solution

The workstation environmental configuration needs to be defined and provided to training engineers prior to the start of training development. If multiple configurations will be used to accommodate the needs of different teams, it is essential that the training workstations have each configuration installed for user training.

3.3 The Software Development/Training Information Gap

Software development consists of specialists who interpret functional requirements from project system engineers, design software to those requirements, and implement the design.

3.3.1 Problems

Software developers interface directly with system engineers who help delineate developing software, but often have little, if any, communication with the users who will be operating it. When software is difficult to learn, end users may seek alternate methods to achieve their processing goals. In a distributed workstation environment, users can (and will) build their own tools to get their jobs done. These tools may propagate throughout the system, spreading by word-of-mouth, without benefit of configuration management or any kind of version control process.

Trainers occasionally have to assume the role of a marketing agent and “sell” users on the capabilities of a software product with a reputation for being too complex. This is an example of why software developers should make sure trainers thoroughly understand and “like” their programs.

Software developers do not always have direct contact with the end user community. They need insight into potential user responses under various conditions. They need someone to test drive the latest iterations of a developing program, someone with a user-like perspective who is willing to provide constructive, “in-house” feedback prior to formal testing.

Training developers do not always get intimate details about what the software is capable of doing, how it processes internally (how it “thinks”), with what subsystems it will interface, what established

processing limits are, what optional processing procedures exist, etc. A trainer cannot teach the functional aspects of a program if he or she doesn’t understand them. Trainers are comfortable teaching what they know well and gloss over things they don’t.

Software frequently contains undisclosed program functions and built-in traits that users may discover before trainers do. These discoveries may weaken a trainer’s credibility.

3.3.2 Solutions

Early access to software products would enable us to begin developing a training strategy.

Pair training developers and software developers. Trainers are often the closest thing to a marketing representative that a software developer may have. Software developers have a vested interest in ensuring that trainers completely understand the logic and functionality of their software products. Understanding the logical sequencing of algorithms will directly affect the trainers ability to answer the “why’s” and “what if’s” invariably asked by students. That knowledge will strengthen a trainer’s credibility with learners.

Trainers also make great user advocates because they possess insight for predicting user preferences and how users will respond to software functionality under given conditions. This insight helps the software developer provide a more usable product.

Training developers must be allowed to “gorilla test” the latest iterations of the software. Early access to program operation makes it possible to develop an early training strategy. Trainers may be able to identify software bugs that could otherwise go undetected prior to testing or even operational use. Early detection provides software developers with an opportunity to “fix” them without failure report processing.

Trainers should be able to validate their developing training modules by lending their presentation skills to developers, assisting them in software performance demonstrations and design walk-throughs.

3.4 The System Administration/Training Information Gap

System administrators (SAs) work directly with users to provide and maintain a distributed system. Training developers are also users, but they rely on the SAs for additional assistance with a variety of special needs.

3.4.1 Problems

We don't know enough about system administration functions, policies, and requirements, or how to design training modules to them.

Training developers may not have access to specific information about each workstation, i.e., what software is available, which versions are installed, what remote access methods are valid, who has access, who "owns" each workstation, who maintains it, and what restrictions are currently in force.

3.4.2 Solutions

Limited cross-training would provide training developers and system administrators some perspective into how the "other half" functions, giving a broader perspective of the system.

A "who's who" map of system nodes would help orient us toward our learners, and a current workstation status report would pinpoint user capabilities.

4. A LESSON IN COMMUNICATION

Prior to the Mars Observer spacecraft launch, we were approached by the Mars Observer Spacecraft Team, which was preparing for its upcoming integration and readiness testing. They needed their engineering personnel trained in workstation operation and downlink processing as soon as possible. The hardware and software installation process was far from complete, but some of their users had workstation access.

Project deadlines were rigid and training short-cuts were required to get them on-line for downlink process testing. We shortened our list of objectives, but the limited seating in our training facility still presented a problem. With two workstations, side-by-side, we could conduct only one hands-on class at a time.

Our alternative was a "housecalls" approach. We could accommodate more students per class by conducting training in their own working environment, using their own workstations. The Spacecraft Team suggested a common area housing six workstations, where all had visibility to a white board, but they were not all within sight of each other.

We decided on the housecalls approach for our "hands-on" participation modules while holding concurrent demonstration overviews in the training

facility. This got users on-line quickly during the initial phase.

The existing user environments on the six workstations determined our next step. Multi-mission Ground Data System workstations run X Windows System software under UNIX with a Motif™ user interface. Our previous customer was the Ulysses project using icon access to files, directories, and processes. We were not prepared for what we found on the Mars Observer Spacecraft Team workstations.

The six Spacecraft Team workstation environments had no common user configuration and, more important, our existing training materials did not resemble the access methods being used. Mars Observer was not using an icon-oriented desktop tool; instead, they preferred a command line interface with pop-up menus. We had presumed that all multi-mission workstation environments would be consistent in their appearance; at least that was the rumor that training received, but these workstations weren't even close.

Although we were familiar with multi-mission software components from the perspective of other projects, we knew very little about Mars Observer, the Spacecraft Team, or the needs of its engineers. All we could promise was a "best efforts" approach to the task. They agreed that workstation consistency was desired, so we went in search of some configuration support.

Training's previous concerns over a multi-mission workstation training configuration had been addressed and responsibility had recently been assigned to our Section's Operations Engineering Laboratory. They had just been incorporated into the new Operations Technology Group and were still attempting to define their responsibilities. Our problems prompted the immediate formation of the Customer Adaptation Team which became a mini-development effort to support the Mars Observer Spacecraft Team and eventually other Mars Observer organizations upon request.

The significance of the Customer Adaptation Team to the training function was our involvement. We provided the impetus for their mobilization and they included us in the development process. Using one of the six workstations as a base of operations, the Customer Adaptation Team propagated its new user configuration to our training workstations and the other five workstations in the Spacecraft Team area. We were able to make practical recommendations that benefited the users and we received each iteration of their configuration to evaluate and use in

designing our new materials. Our materials become the documentation for their development effort. The new configuration included pop-up menus and graphic user interface displays. They also provided a controlled directory structure for storing the large numbers of project-controlled scripts and required process support files that had previously been elusive entities without version protection.

Our first Spacecraft Team class was held without time for a dry run, so the first day was a debugging session. Only two users had accounts, and those accounts were not valid on all machines; some of the disks were full so that whenever we encountered a program that logged information to a file, it would "hang," etc. Without direct eye contact, it was difficult to communicate over the noise of the machines and there were numerous distractions, but at the end of the day our students' response was positive. Each day went more smoothly than the last, and we proved to ourselves that housecall training is an option worthy of serious consideration.

The most meaningful result was the obvious convenience to users. There was no transition; the learning environment became their working environment. Every workstation they were authorized to use contained the same functional configuration. *The point is that consistency is essential to reinforce learning.*

5. CONCLUSIONS

End users depend on training to lead them through the learning experience with dignity. They expect trainers to understand all aspects of the software and the processing methods necessary to accomplish and expedite their processing responsibilities. They expect trainers to translate the capabilities of the software into performance methods for accomplishing their tasks. They expect practical responses to their "how do I get from here to there" questions, even if it means providing a work-around for their specific needs.

For training to benefit the end user, it needs to be a quality and timely experience. It needs to be available when the user needs it, and the best way for that to happen is for a trainer to be a sponge for knowledge, a conductor of information, and an advocate for the end user during the development cycle. Software trainers need to experience the software, first-hand, before it hits the streets. They need to become intimately familiar with the inner workings of the software and not simply exposed to a "hand-is-quicker-than-the-eye" public demonstration.

Trainers have a unique perspective on the functionality of software in the hands of end users, and can provide insight during the design process when changes are inexpensive and easy to implement. Pairing training engineers with development engineers benefits both. Trainers gain the experience and insight into the product necessary to begin early development and testing of training materials. Programmers gain a resource with a users' point of view for dry-running early versions and providing feedback on usability. Trainers are experienced presenters and can assist in demonstrations and overviews that illustrate software development status.

6. RECOMMENDATION

Strive for a single "package delivery" for software, training, and documentation products based on functional requirements. Cross-communication provides consistency among software, documentation, training, and end user needs.

The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

THE COLUMBUS LOGISTICS SUPPORT AT THE APMC:

REQUIREMENTS AND IMPLEMENTATION ASPECTS

N94-23935

C. Canu
Agenzia Spaziale Italiana (ASI)
Roma (Italy)

L. Battocchio
ALENIA SPAZIO S.p.A.
Torino (Italy)

S. Masullo
CISET S.p.A. - Space Division
Roma (Italy)

ABSTRACT

This paper focuses the logistics support to be provided by the APM Centre (APMC).

Among the Columbus ground infrastructures, this centre is tasked to provide logistics, sustaining engineering and P/L integration support to the ongoing missions of the APM, i.e. the Columbus Laboratory attached to the Freedom Space Station.

The following is illustrated:

- an analysis of the requirements that are levied on the logistics support of the APM;
- how such requirements are reflected in the corresponding support to be available on-ground and at APMC;
- the functional components of the APMC logistics support and how such components interact each other;
- how the logistics support function interfaces with the other functions of the ground support;
- how the logistics support is being designed in terms of resources (such as hardware, dataware, etc...).

Emphasis is given to the data handling aspects and to the related data bases that will constitute for the logistics activities the fundamental source of information during the APM planned lifetime.

Functional and physical architectures, together with trades for possible implementation are addressed. Commonalities with other centres are taken into account and recommendations are made for possible reuse of tools already developed in the C/D phase.

Finally programmatic considerations are discussed for the actual implementation of the centre.

Keywords : APM Centre, Columbus, Ground Segment, IOI, Logistics.

1. INTRODUCTION

Columbus will be undoubtedly a demanding challenge for Europe in the next decades.

For the first time Europe will deal with crew permanently in orbit and with space elements to be serviced and maintained operational for many years.

As a matter of fact thirty years of uninterrupted operations of orbital elements are a challenge not only for the mission operational aspects but also for the whole of the ground infrastructures entrusted to support the space elements.

All the on-ground functionalities requested by the Columbus missions are implemented by the facilities of the IOI (In-Orbit Infrastructure) Ground Segment. Such functions can be grouped basically as follows:

- Mission coordination & planning;
- Monitoring & control of the space elements;
- Communication capability;
- Crew training;
- Integration & Logistics Support.

These functions are all essential for a safe and cost effective mission objective achievement.

A critical aspect, both technical and organisational, will be constituted by the effort to maintain both the space and ground segment during all their planned life cycle.

Consequently logistics will play a fundamental role in the Columbus era.

For the accomplishment of most of the Columbus APM logistics tasks, a new centre has been introduced inside the IOI ground infrastructures, that implements basically new operational concepts w.r.t. the previous space experiences.

It is the APMC, tasked to provide logistics, sustaining engineering and P/L integration support to the APM missions.

2. NOMENCLATURE

APM	= Attached Pressurised Module, called also Columbus Attached Laboratory
APMC	= Attached Pressurised Module Centre
ASI	= Agenzia Spaziale Italiana
CMCC	= Central Mission Control Centre
DB	= Data Base
DP	= Data Processing
EAC	= European Astronaut Centre
EGSE	= Electrical Support Ground Equipment
ILS	= Integrated Logistics Support
IOI	= In Orbit Infrastructure
MSCC	= Manned Space Lab. Control Centre

ORU = Orbit Replaceable Unit
 PHS&T = Packaging, Handling, Store & Transport
 P/L = Payload
 POIC = P/L Operation & Integration Centre
 SSCC = Space Station Control Centre
 UHB = User Home Base
 USOC = User Support & Operation Centre

3. REQUIREMENTS OF THE APMC

The aims of the APMC are:

- To provide a repository of knowledge and expertise concerning the APM in order....
-to support the Space Station Freedom Control Centre and the Columbus Control Centres in operating the APM in orbit and maintaining it safe and operational as required...
- ...and to support users of the APM in designing, preparing and operating payloads.

In order to reach its aims, the APMC has the following main roles:

- as a referral centre for all other centres involved in the management and operation of the APM;
- as a support centre for potential and actual payload users;
- as a resource which may be used by the SSCC and/or the crew to help resolve contingency situations.

The centre will be responsible for many facets of the support to be provided both to those operating the APM and to its users, i.e. those who wish to fly payloads on it. The result of such a broad remit is the broad range of activities the centre must undertake. These range from writing software for the APM to arranging intercontinental transport for equipment and spares.

In order to fulfil the demanded tasks, the APMC must have many external interfaces, that, as regards the ground infrastructures, are summarised in figure 1.

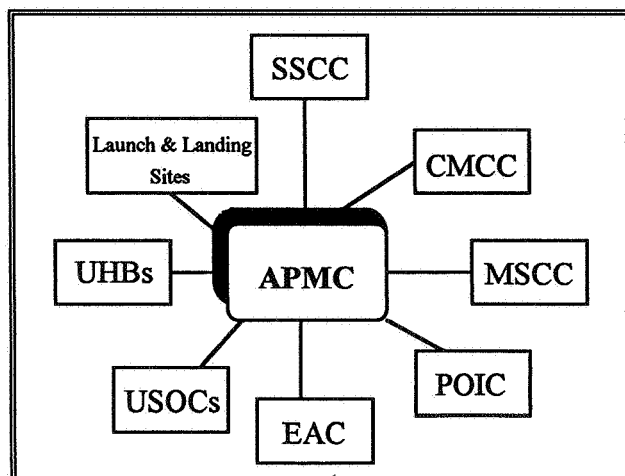


Fig. 1 APMC major interfaces with ground centres

All the activities that are required to be performed in the centre can be grouped in the following major functions:

- Engineering support
- Payload Integration Support.
- Integrated Logistics Support

The *Engineering Support* encompasses all the activities necessary to support the system engineering mission operations throughout the APM lifetime, i.e.

- Flight increment preparation support;
- Mission control support during operations execution;
- Acquisition, processing and distribution of telemetry data;
- High fidelity crew training;
- Configuration control;
- Strategic and tactical planning support;
- Operations assessment;
- Maintenance of flight configuration on board software and data bases;
- Maintenance of the board flight data files;
- Data archiving;
- Engineering assessment of design change proposals;
- Maintenance of the Engineering Data Library and Data Bases.

The *Payload Integration Support* activities envisioned in both the pre-operational and operational phases of the program include the establishment and maintenance of a capability that supports the decentralised user payload development and integration concept. It will be responsive to the User community requirements throughout the lifetime of the program. This function will provide for:

- Payload development and integration planning;
- Payload verification;
- Launch site payload processing and carrier integration;
- On-orbit operations monitoring;
- Landing site carrier de-integration and payload processing;
- Payload integration support function management and coordination.

The *ILS* includes the establishment and maintenance of an integrated support system that will be responsive to the flight element and program requirements for logistics processing.

The related functions are detailed in the subsequent section, devoted to the description of the logistics activities the APMC has to deal with.

Such major functions do not operate as stand-alone activities, but need a mutual and continuous exchange of data, plans and information with a frequent use of same equipment in order to best accomplish their assigned tasks.

4. LOGISTICS AT APMC

The logistics expected to be performed at APMC can be basically split into two groups of activities, i.e. :

- logistics in support of the APM, that is the system in orbit, to be supported effectively during its planned life;
- logistics supporting the centre itself.

Such a splitting is only functional and does not prevent that same operational equipment and personnel implement functions belonging to both of the a.m. groups.

As a matter of fact the basic functions that compose the APMC ILS capabilities can be applied to both APM and APMC as shown in the figure 2.

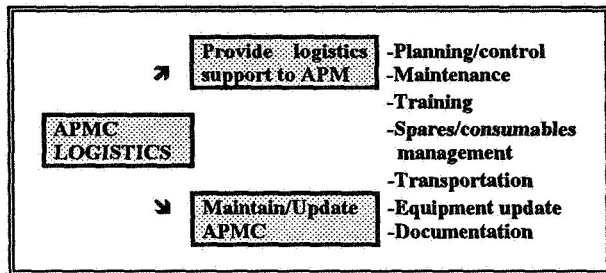


Fig. 2 APMC Logistics functionalities

These functions are detailed in the next following lines.

As a matter of fact the totality of the logistics tasks at the APMC can be well described as follows:

- Maintenance of all the ground support, airborne support and flight hardware at the organisational, intermediate and depot levels, including performance of internal and overhaul activities and verification and control of the outside contracted ones.
- Technical support of in orbit maintenance and servicing via situational analysis, simulation and interdisciplinary coordination with other centre functions. This is to aid in anomaly resolution during both nominal and contingency maintenance and servicing.
- Supply support for the maintenance function in terms of providing spares and repair parts, warehousing, inventory management and control, re-procurement, supplier and vendor retention, repair and overhaul process support and control.
- Transportation services for the maintenance and supply support items to and from their respective sources and user locations.
- Training for those personnel required to carry out the maintenance and supply support functions of logistics and support to crew training for the in situ repair procedures.
- Ground processing for the resupply and return hardware and those items of operational ground support equipment that are required to support the flight segment of the program.

- ILS-used documentation for the support of the maintenance, supply support, training and transportation including design and programmatic changes that effect those functions.
- Facilities management for the operation and maintenance required to house and support the maintenance, supply support, training and transportation functions.
- Limited logistic engineering capability for:
 - Monitoring and participation to design requirements modification;
 - Participation to the generation of new requirements and related specification;
 - Logistics issues monitoring and control for new APM and/or APMC items/equipment design and development contracted outside the APMC.
- Operational management, control and integration of:
 - ILS functions;
 - Their interfaces and coordination with the engineering support and payload integration support functions;
 - Strategic and tactical planning for flight increment manifest planning, including nominal and rescheduled flights with ILS impacts.

At the beginning of APM and APMC operations, the logistics activities shall be mainly based on suppliers inputs, related to MTBF, reliability parameters, costs etc. and on simulations.

As more information becomes available on system and subsystems due to the analysis of live data coming from APM and APMC operations, the logistics programs shall be upgraded in order to improve the logistics support in terms of minimising costs and maximising efficiency of all the system (i.e. space and ground).

5. LOGISTICS FACILITIES

The analysis conducted in the previous sections addressed only the functional aspects of the centre and in particular of those functions affecting the logistics to be performed in the centre.

All the identified functions must be translated into a set of facilities able to implement the needed functions. As regards the APMC logistics facilities, these can be grouped essentially in:

- Data Bases;
- DP-based facilities;
- Training materials and tools;
- PHS&T tools, materials and facilities;
- Workshops.

The figure 3 shows in a pictorial way the components of such facilities and simultaneously aims at providing the reader with the importance of their mutual

interactions for an effective conduction of the APMC operations.

It is also worthwhile mentioning the fundamental role the data bases have to play in the APM and APMC life, in support of most of the logistics facilities of the centre as intentionally pointed out in the figure.

It is beyond the scope of the paper to detail these facilities; only an overview is provided in order to show how the a.m. requirements can be satisfied by such facilities.

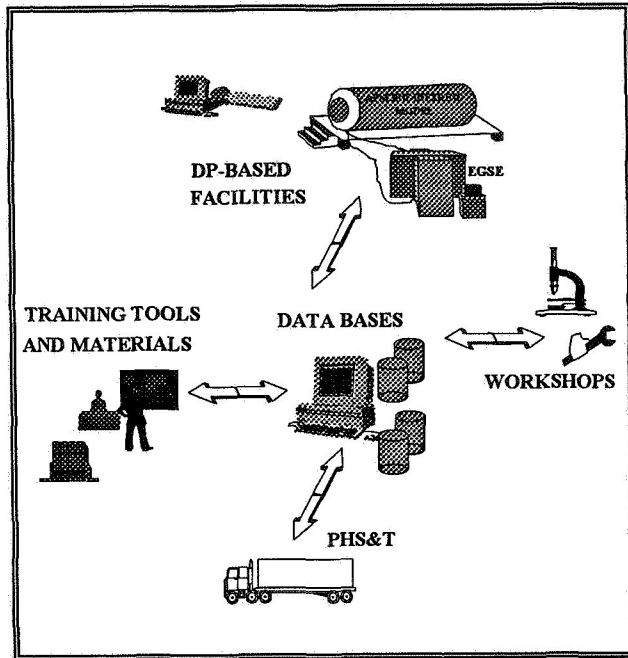


Fig. 3 Major APMC logistics facilities

5.1 Data Bases

The APMC will use several kinds of databases necessary during the performance of APM operations. Taking into account the ones that have major impact on the logistics activities, they can be summarised as follows :

- Mission data bases;
- Engineering D/B;
- Logistics D/B;
- Other Preparation Support D/B;
- APMC archives.

The figure 4 shows these groups of major data bases planned to be used at the APMC and the role of the logistics DB (it is not an architecture but only a picture aiming at summarising the APMC DBs and their interactions).

The Mission data bases encompass all the data related to the APM increments.

The data are organised per increment in order to allow the preparation of more than one increment at the same time during the preparation phase.

For each increment this D/B encompasses:

- Data related to APMC configuration;
- Data related to APM configuration;
- Payload data references;

- Fault management data;
- Crew reference documentation.

The Engineering D/B embraces all the needed engineering data for APM, APMC and P/Ls, i.e. :

- graphical data, such as drawings, schemes, etc.;
- text data (such as requirements, crew procedures, flight data files, etc.);
- references to video and audio recordings;
- reference to paper documentation.

These engineering data are in many different forms, created by many diverse applications, not all of them necessary computer resident.

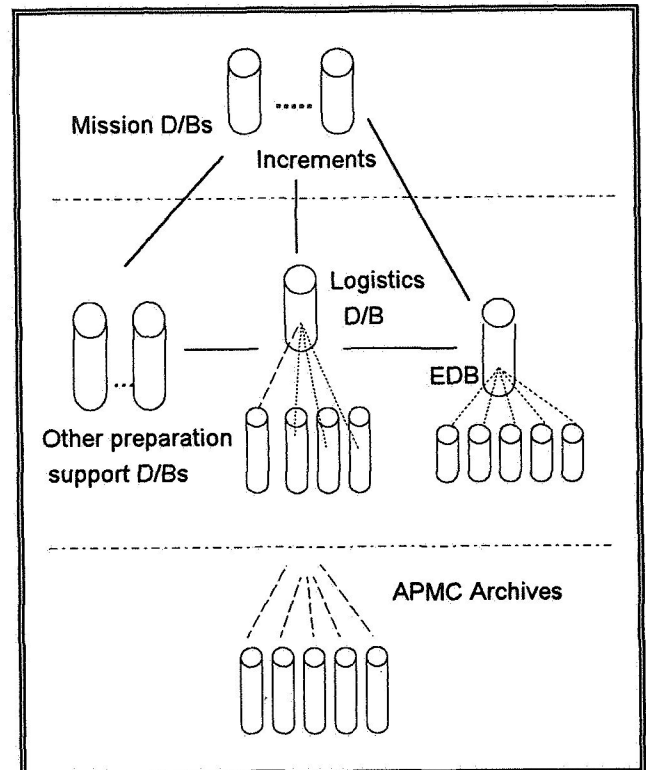


Fig. 4 DBs role at APMC

The Logistics D/B contain all relevant logistics data, classified as follows:

- Engineering data, encompassing
 - RAM data;
 - MTTR and MTBF data;
 - FMECA data
 - Others
- PHS&T data, related to
 - Packaging/Unpacking data and instructions;
 - Spares relevant data (size, weight, storage conditions);
 - Commercial data;
 - Others
- Training data, concerning
 - Skills;
 - Training results;
 - Etc...
- Documentation data;
- Historical data, mainly related to the results of maintenance interventions. Such data will be also

useful to analyse effectiveness of logistics organisations of the centre and to correct possible unfulfilment of imposed requirements of Availability and Reliability.

Other preparation support data bases are constituted by planning , P/L and on-board D/B.

The APMC Archives are made up by all data used during the execution of all the APM increments, encompassing the current increment and the data related to the past increments.

The complexity of such tools among APMC facilities will be adequately reflected in the H/W and S/W architectures with the related interfaces to both internal and external networks in order to get an efficient transferring of all the needed data during the execution of the logistics operations among the facilities (as shown in fig. 3) concerned.

5.2 DP-based facilities

The logistics activities at the APMC will exploit facilities, that heavily rely on data processing systems. An important set of tools that will be used in support of activities of the APM will be surely constituted by the APM Engineering Model and the related EGSE. It will be used to provide the capability to verify functions and measure performances for the purpose of acceptance, trouble-shooting and repair.

A more flexible tool is the APM Simulation Facility, that will be taken over from the C/D phase of the APM flight segment. Unless the breadboard of the on-board Data Management System (DMS) it is completely S/W-based and will allow prompt simulated configuration of the APM to help assess specific logistics operations, when needed.

Other DP-based tools are constituted by S/W programs running on workstations and/or PCs for specific purposes (such as logistics planning, reliability analysis tools and others).

5.3 Training materials

The training here addressed is only related to the personnel in charge to perform the logistics tasks as described in the previous section 4.

Anyhow commonalities of tools and facilities is expected among the diverse training tasks to be performed in order to optimise schedules and costs.

The planned materials can be summarised as follows:

- classrooms
- workbooks
- computer aided instructors (CAI)
- presentation materials
- mock-ups
- simulators
- instructor training materials
- tools to support training verification process

It is to point out that other training materials are constituted by the actual APMC operational

equipment, since on the job training is expected to be conducted as much as possible in the centre.

5.4 PHS&T tools

The APMC will have stores, highly equipped and automated to allow effective movement, handling and distribution of materials, supplies and equipment in the centre and outside.

Tools encompassed in such a category are constituted by lifting devices, cranes and facilities to help keep track of items under APMC control (such as bar code readers, optical devices, etc...) in order to get an optimum stocking of items.

5.5 Workshops

Equipped workshops will allow to perform all the intermediate maintenance of the centre and to carry out tasks in support of the flight element and the related payloads.

Workshops allowing to intervene in electrical and electronic, mechanical and fluidic works are consequently necessary in the APMC.

These workshops shall be equipped with off-the-shelf tools (such as multimeters, frequency generators, etc...) plus ad-hoc facilities (e.g. workbenches to execute peculiar maintenance on specific component of the centre).

5.6 Commonalities and trades

Most of the facilities needed inside the centre to support the logistics operations don't have to be developed ad-hoc.

These could be grouped in two main categories:

- already developed for the space segment and reused, after due modification and/or customisation in the centre (such as DBs developed during the C/D phase);
- common with other ground facilities (such as training materials).

The development and/or transfer of such facilities must be phased carefully with the actual needs of the diverse users in order to have on time the requested support.

Furthermore trades will be necessary to best assess options in the off-the-shelf equipment.

6. PROGRAMMATIC ASPECTS

The APMC, developed by ESA for the above mentioned functions, will be located in an ASI infrastructure, being designed by Alenia Spazio in the frame of a national contract covering all the buildings and the general support services definition.

Parallel and complementary ESA and ASI studies (both implemented by Alenia Spazio as Prime, with Ciset contributions) are running in order to complete the design of the entire facility in all its own features.

The Italian portion of the study will be over within the year 1993 and the planning is to start the building-up of the infrastructures at the beginning of 1994.

The physical facility should be completed two years later with all the internal general services equipment ready to support the ESA APM mission specific infrastructures and staff.

In addition to the above mentioned planning it is important to mention, for a clear general picture on APMC what are the programmatic aspects from ASI side on the Centre.

ASI is involved with NASA in the development of the Space Station Freedom Logistics Modules. At the moment the studies on the first module, the so called Mini-Pressurised Logistics Module (MPLM), are entering in phase C/D.

ASI is planning, and negotiating with NASA, to host facilities acting for different functions in support to the Logistics Modules.

Those functions (like sustaining engineering, logistics support, some integration tasks, etc.) shall be managed by an Italian Centre in strict coordination with the NASA facilities.

ASI plans to develop this Centre in the same location of the APMC (city of Torino area) using the same general support services already planned for the latter, in full synergy for all the common aspects and with the goal to improve the final total performance of the entire system versus each single program needs.

7. CONCLUSIONS

The importance of logistics activities in their multiple aspects for a long-duration space programs, such as Columbus, has been addressed, showing the importance of the APMC in the ground infrastructures in support of the APM operations.

The building of such a centre should prove again that ILS has to be performed not as a stand-alone and a post-development activity, but has to be conducted in parallel with the implementation phase of the centre, keeping continuous touch with the C/D phase running activities of the flight element.

More embedded is ILS in the overall APM program and more effective will result in the overall life of the entire system.

Due to the peculiarities of the Columbus program, APMC is expected to play a vital role in the operational phase of the APM with its available logistics infrastructures, for a cost effective accomplishment of the APM targets.

8. ACKNOWLEDGMENTS

The authors wish to thank Dr. G.F. De Luca for the technical advice provided in the preparation of the paper.

9. REFERENCES

1. Battocchio L., Harris D.R. and Sarlo L. **The Columbus APM Centre Flexible and Efficient Engineering Support.** ESA SP-308, October 1990.
2. De Luca G.F., Masullo S. and Randisi S. **The Requirements on Data Systems of Columbus Logistics and Engineering Support.** ESA SP-308, October 1990.
3. De Luca G.F., Grazi R. and Masullo S. **Integrated Logistics Support: Suggested Approach for Columbus.** 3rd Space Logistics Symposium, Colorado Springs, April 90.
4. Masullo S. and Berni R. **Real Time Simulation Approach in Columbus Pressurized Module.** International Symposium on Space S/W Engineering, Torino, December 1988.

N94-23936

AUTOMATING A HUMAN FACTORS EVALUATION OF GRAPHICAL USER INTERFACES FOR NASA APPLICATIONS: AN UPDATE ON CHIMES *

Jianping Jiang Elizabeth D. Murphy
 Sidney C. Bailin
 CTA INCORPORATED, Rockville, MD

Walter F. Truskowski
 NASA-Goddard Space Flight Center, Greenbelt, MD

ABSTRACT

Capturing human factors knowledge about the design of graphical user interfaces (GUIs) and applying this knowledge on-line are the primary objectives of the Computer-Human Interaction Models (CHIMES) project. The current CHIMES prototype is designed to check a GUI's compliance with industry-standard guidelines, general human factors guidelines, and human factors recommendations on color usage. Following the evaluation, CHIMES presents human factors feedback and advice to the GUI designer. The paper describes the approach to modeling human factors guidelines; the system architecture; a new method developed to convert quantitative RGB primaries into qualitative color representations; and the potential for integrating CHIMES with user interface management systems (UIMS). Both the conceptual approach and its implementation are discussed. This paper updates the presentation on CHIMES at the first International Symposium on Ground Data Systems for Spacecraft Control[1].

Key Words: GUI, human factors guidelines, GUI design style guidelines, knowledge representation, color.

1. INTRODUCTION

An effective user interface (UI) is key to the success of any complex, automated system, such as a ground control center for spacecraft operations. Many efforts have been made to assist user-interface designers in building usable visual displays. User-interface prototyping tools provide direct interaction between the designer and the

design, as well as reduced programming through automatic code generation[10]. Although building a user interface requires less effort than ever before, many poor designs can still be found because they violate human factors design principles or industry-standard guidelines, such as the specific requirements specified in the *OSF/Motif Style Guide*[9]. Improving poor UI designs requires human factors expertise and style-guidelines knowledge, which are, however, expensive and not readily available during design and development. Automated human-factors-guidelines and style-guidelines compliance checking are promising approaches to the problem[8].

Under the direction of NASA-Goddard Space Flight Center, a series of research and prototyping cycles has produced the current fourth-generation CHIMES methodology and toolset[7]. A previous paper describes the original CHIMES methodology and early prototypes[1]. The current GUI-evaluation prototype includes a style-compliance-checking tool, i.e. an automated critic that checks a design against style rules implemented in the tool's knowledge base. This prototype includes capabilities to evaluate conformance to human factors guidelines and recommendations on color usage. The tool provides feedback within the context of identified instances of non-compliance and advises the GUI designer on how to modify the design.

In the sections that follow, we describe the implementation approach used in the current prototype, the architecture, the implementation of compliance-checking rules, and plans for future research. Color representation in the knowledge base is also described.

*This work was performed at the NASA-Goddard Space Flight Center (Code 522.3) under Contract Number NAS5-30680, with the support of NASA Headquarters (Code O).

2. IMPLEMENTATION APPROACH

Human factors guidelines cover many aspects of a GUI design, from color selection to screen layout. Style guidelines provide more specific requirements. For instance, the *OSF/Motif Style Guide* requires a menubar to be placed at the top left of a screen. Because so many factors are involved, a GUI design's non-compliance with human factors guidelines or style guidelines can take many forms. In most cases, the problems of non-compliance are independent. Sometimes, however, they are inter-connected. For example, using smaller fonts may cause poorer legibility of colored symbols. Thus, checking non-compliance problems in GUI designs demands consideration of many factors and diverse responses from a compliance-checking program. This requires an effective implementation approach to support the complex control of switching between blocks of code.

The conventional programming approach, used by systems written in C or FORTRAN, fails to provide adequate support, because the code in such a system is executed following a predetermined sequence, and a programmer is required to develop the control structure, which is very complex in the case of non-compliance checking. A more appropriate approach is a rule-based system using forward chaining, because its control mechanism automatically executes the right blocks of code based on input data.

A rule-based system has three components: a data memory, a production memory, and an inference engine[3]. Data memory stores the input data and the current state of knowledge during the problem solving process. Production memory stores the rules which constitute the program. The inference engine executes appropriate rules based on the configuration of data memory. A rule has two parts, a conditional part and an action part. The conditional part describes the data memory configuration for which the rule is appropriate, and the action part gives the instruction for changing the data memory. During code execution, rules that are satisfied by the current content of data memory are collected; one of the collected rules is then selected based on some selection strategy; finally, the selected rule is executed.

When forward chaining is used in rule-based systems, rules are executed whose conditional parts are satisfied by the current content of data memory. The rules executed are determined by in-

put data because the content of data memory is a transformation of input data. This frees programmers from writing complex control code, thus providing better support for system development.

By using forward chaining, a compliance checker can represent GUI designs as facts in the data memory and can model compliance-checking expertise in the form of rules which match non-compliant patterns. When a GUI design violates the guidelines, the facts in the data memory will reflect the violation, which will match a condition of the compliance-checking rules. This leads the inference engine to execute the appropriate program code.

3. CHIMES ARCHITECTURE

The current CHIMES prototype architecture, illustrated in Figure 1, uses a rule-based-system-with-forward-chaining approach to checking a GUI's compliance with human factors guidelines and style guidelines. There are five modules: design acquisition module, compliance-check controller, knowledge base, advice generator, and user interface.

The purpose of the design acquisition module is to acquire GUI design information. It transforms a design into input data for CHIMES to evaluate. This module consists of two submodules, the automatic design capture submodule and the manual design capture submodule. The automatic design capture submodule acquires GUI design information by reading a design description file, such as a TAE¹ resource file. The manual design capture submodule supplements the automatic design capture submodule by capturing additional design information, through queries to the GUI designer.

The primary function of the compliance-check controller is to initiate a CHIMES evaluation of an acquired design. It takes the design information from the design acquisition module, initializes the knowledge base, converts the information into facts, asserts the facts into the knowledge base, activates compliance checking rules, and initiates the evaluation.

The knowledge base is a key component of the CHIMES prototype system. It contains the rules

¹TAE is a trademark of the National Aeronautics and Space Administration. The Transportable Applications Environment (TAE) Plus is a user interface development and management environment[10].

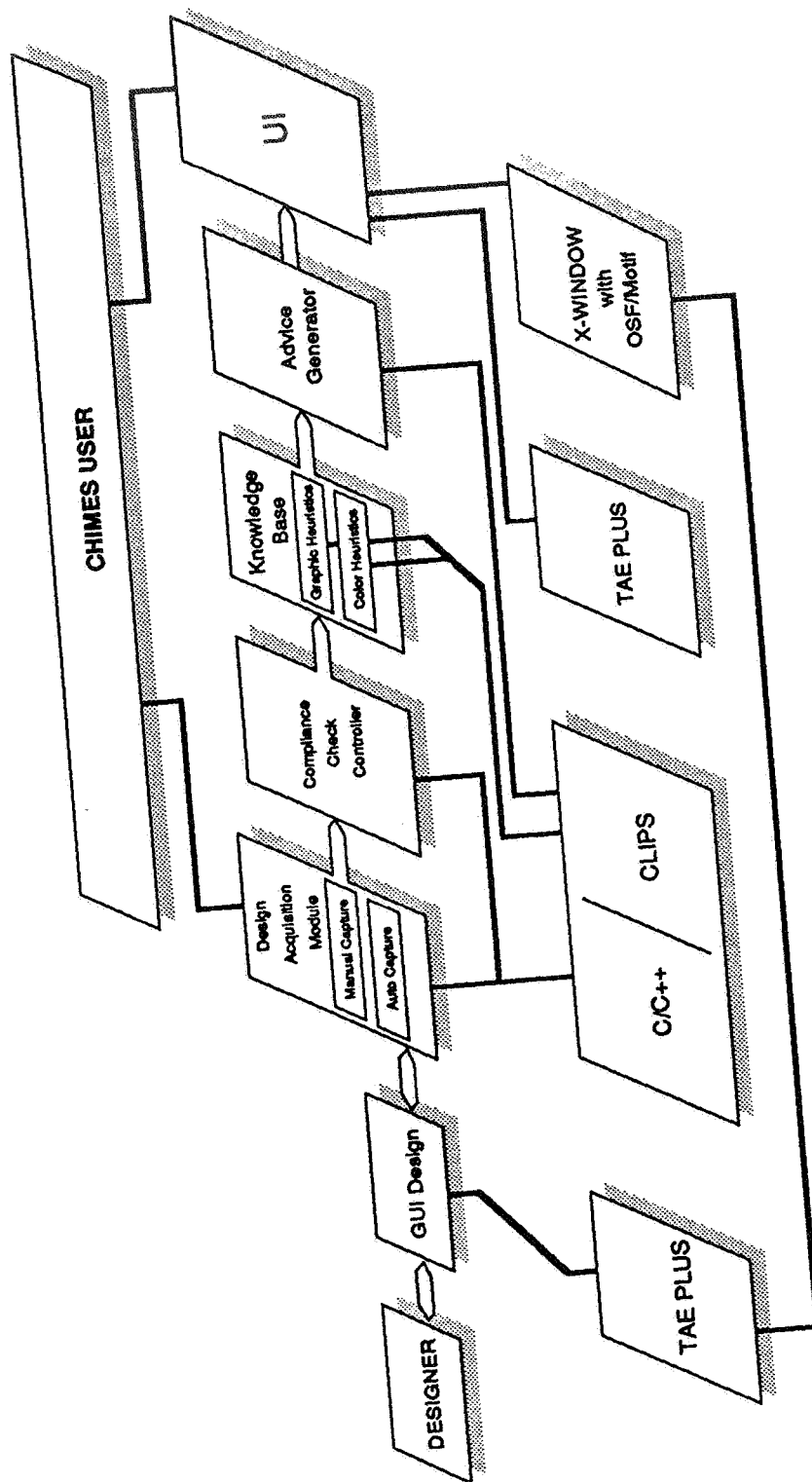


Figure 1: CHIMES Architecture

that model compliance-checking expertise and the facts that represent the user-interface design being evaluated. Upon receiving the evaluation instruction from the compliance-check controller, the knowledge base performs the compliance check and calls the advice generator to generate applicable advice.

The advice generator presents the results of compliance checking to the GUI designer. It prepares advice based on the results of compliance checking. The advice is then displayed in the CHIMES user interface, with an indication of the display element that triggered the advice. This provides advice in the context of the problem.

The user interface allows the GUI designer to initiate compliance checking. It also serves to present the compliance checking results to the GUI designer. For ease of use, all user interactions in the prototype system occur via mouse positioning and clicking.

The knowledge base is implemented by using CLIPS²[4], which supports development of a rule-based system with forward chaining. The user interface is developed by using TAE Plus with X Window System³ and OSF/Motif⁴. The design acquisition module, the compliance check controller and the advice generator are written in C and C++.

Interaction of Modules. Compliance checking begins with the design-acquisition module, which currently acquires a user-interface representation from a TAE resource file and transforms it into input data for the CHIMES evaluation. Once the GUI representation has been captured, it is sent to the compliance-check controller, which issues requests for compliance checks to the CHIMES knowledge base. If violations of the guidelines are found, advice is generated on how to improve the design.

4. IMPLEMENTATION OF RULES

Checking a GUI design's compliance with human factors guidelines and style guidelines is performed

²CLIPS (C Language Integrated Production System) is a trademark of the National Aeronautics and Space Administration.

³X Window System is a trademark of the Massachusetts Institute of Technology.

⁴OSF/Motif is a trademark of the Open Software Foundation.

in the knowledge base, where a design is represented as facts in the data memory, and the compliance-check expertise as rules in the production memory. This section discusses the implementation of compliance-check expertise in the knowledge base.

CHIMES' compliance-checking rules are developed by modeling non-compliance patterns. These patterns take various forms, such as geographic violations; for instance, placing a menubar at the bottom of the screen violates the OSF/Motif style guidelines. Overuse of design elements, for instance, using five fonts in a screen or color mismatches, for instance, choosing black for the background color and red for the foreground color, can yield poor legibility or visual discomfort[11].

The non-compliance patterns are described in the conditional parts of the rules, which catch the violations. For example, a rule examining menubar position has the following form:

```
(defrule check-menubar-location
(goal-is check-menubar-location)
(item (name ?name) (type menubar))
(loc (name ?name) (x ?x&:(≠ 0 ?x))
(y ?y&:(≠ 0 ?y)))
⇒
(advice "According to OSF/Motif Style
Guide, the menubar should be placed at the
top left of the window."))
```

The conditional part of this rule defines three conditions for the rule to be executed: 1) the knowledge base is checking menubar location; 2) there is a menubar item in the data memory; and 3) the menubar is not located at the upper left corner of the screen. The action part of the rule specifies which piece of advice should be generated if a non-compliant menubar is found. This rule catches any violation of standard menubar placement, which may have the following form in the data memory:

```
(item (name mb) (type menubar))
(loc (name mb) (x 170) (y 10))
```

While many of the violation patterns can be represented in the knowledge base directly, color is an exception, because the form of color representation used for electronic display is not appropriate for evaluation by rule-based systems.

Color is produced on an electronic display by additive mixture of three primary colors, red, green

and blue. Each primary color is associated with a number to indicate its intensity. For instance, yellow is represented in OSF/Motif as (255, 255, 0), indicating that yellow is the mixture of red and green.

A prominent problem that prevents use of this representation in rule-based systems is the size of the color space, which makes developing rules a formidable task. For example, a color space with each primary color ranging from 0 to 255 consists of more than sixteen million colors. It would require more than sixty trillion rules to cover all the possible color combinations.

To reduce the size of color space, a linguistic color model, the color-naming system (CNS)[2], is used in CHIMES. The CNS model consists of about three hundred color names. Each name contains information on hue, saturation, and lightness. An example of a CNS name is "*very light vivid yellow*," where the saturation modifier *vivid*, and the lightness modifier, *very light*, supply more information on the hue *yellow*.

To use CNS color names in the knowledge base, color representation conversion is needed, because most computer systems use red-green-blue primary colors to represent color. The conversion involves three color models, the red-green-blue (RGB) model[5], the hue-saturation-value (HSV) model[11] and the CNS model. First, a color is translated into the representation of the RGB model, which uses red-green-blue primary colors but with the intensity range between 0 to 1. Second, a color's RGB representation is translated into the HSV representation using a transformation formula[12]. By using the HSV representation, a color's hue is specified in terms of degree on a color ring; saturation and value, or "lightness" which indicates color intensity, are described within the range from 0 to 1. At last, the HSV representation is mapped into a CNS color name by using a mapping matrix. For example, A yellow, represented as (255, 255, 0) in OSF/Motif, can be translated into the RGB representation (1, 1, 0). Then, the RGB representation can be translated into an HSV representation (60, 1, 1), which subsequently can be mapped into a CNS name, *very light vivid yellow*, because the 60th degree hue is mapped to yellow on the color ring, and the high lightness and saturation map to the modifiers, *very light* and *vivid*.

When checking a GUI design's compliance with

human factors guidelines on general color usage, color representation conversion is done by the compliance-check controller. It translates all the colors in the design into CNS representation, and stores them in the data memory, where they may trigger compliance-checking rules.

5. FUTURE RESEARCH PLANS

Automated compliance checking is a promising approach to developing a utility in UIMS, such as NASA's TAE Plus. Such a utility can help a GUI designer build more usable user interfaces. It can extract information from the system resources; evaluate the design against guidelines; and provide advice to the GUI designer on how to improve the design. The GUI designer can then use user-interface building capabilities in the UIMS to correct design flaws and produce a better user interface design. The current CHIMES prototype is our first step towards developing such a utility. Future plans include expanding the compliance-checking capability, developing a generic input interface, and creating a style guidelines description language.

The scope of a compliance-checking capability is key to its success. Currently, the CHIMES compliance check is limited to one screen and to the "look," i.e. the visual aspect, of a GUI design. We are going to expand the CHIMES compliance-checking capability so that it can evaluate a design with multiple screens. To lay the foundation for achieving this objective, we plan to develop heuristics on consistency of layout, labeling, and color usage. We also plan to develop heuristics on evaluating the "feel," i.e. the behavioral aspect, of a GUI design, which requires recording a GUI's behavior during execution time.

We plan to expand the CHIMES design-input capability. Currently, CHIMES extracts design data from a TAE resource file. An interface to a standard user interface description language, such as UIL[6], will make CHIMES capable of evaluating GUI designs developed in other UIMS environments.

A style guidelines description language is needed when developing a utility to evaluate GUI designs against various style guidelines. The idea of evaluating GUIs against various guidelines is supported by an existing CHIMES feature, that the designer can change the CHIMES compliance-checking behavior by changing its rules in the knowledge base

without needing to re-compile. With an adequate guidelines description language and a translator that translates guidelines into compliance-checking rules, a compliance-checking utility will be able to check a design's compliance with style guidelines such as those developed for OSF/Motif, Open Look, or other customized toolkits.

References

- [1] Bahder, S. A. Murphy, E. D. Sheppard, S. B. & Truszkowski, W. (1990). Developing a user-interface-evaluation tool for space-station-era applications. *Proceedings of the International Symposium on Ground Data Systems for Spacecraft Control* (ESA SP-308, pp. 583-587). Paris: European Space Agency.
- [2] Berk, T., Brownston, L., & Kaufman, A. (1982). A new color-naming system for graphics languages. *IEEE Computer Graphics and Applications*, 2, 37-42, 44.
- [3] Brownston, L., Farrell, R., Kant, E., & Martin N. (1985). *Programming expert systems in OPS5*. New York: Addison-Wesley.
- [4] *CLIPS Reference Manual (Version 5.0)*, (1991). Houston, TX: Software Technology Branch, Lyndon B. Johnson Space Center.
- [5] Foley, J., van Dam, A., Feiner, S., Hughes, J. (1990). *Computer Graphics*. New York: Addison-Wesley.
- [6] Heller, D. (1991). *Motif programming manual*. Sebastopol, CA: O'Reilly & Associates, Inc.
- [7] Jiang, J., Murphy, E. D., & Bailin, S. C. (1992). *Comprehensive CHIMES report (Prepared for NASA-Goddard Space Flight Center, Code 522.3)*. Rockville, MD: CTA INCORPORATED.
- [8] Jiang, J. Murphy, E. D. Bailin, S. C. & Truszkowski, W. (In press). Prototyping a knowledge-based compliance checker for user-interface evaluation in Motif development environments. *Proceedings of the Second Annual World Conference On Motif Application Development and Use*.
- [9] Open Software Foundation, (1991). *OSF/Motif style guide (Revision 1.1)*. Englewood Cliffs, NJ: Prentice-Hall.
- [10] Szczur, M. R. (1992). NASA's TAE Plus: A GUI development tool and application environment. *The X Resource*, 2, 56-77.
- [11] Thorell, L. G., & Smith, W. J. (1990). *Using computer color effectively*. Englewood Cliffs, NJ: Prentice-Hall.
- [12] Travis, D. (1991). *Effective color displays: Theory and practice*. New York: Academic Press.

SECURITY ASPECTS OF SPACE OPERATIONS DATA

Stefan Schmitz

TÜV Rheinland, Cologne, Germany

N 94 - 23937

ABSTRACT

This paper deals with data security. It identifies security threats to European Space Agency's (ESA) In Orbit Infrastructure Ground Segment (IOI GS) and proposes a method of dealing with its complex data structures from the security point of view. It is part of the "Analysis of Failure Modes, Effects Hazards and Risks of the IOI GS for Operations, including Back-up Facilities and Functions" carried out on behalf of the European Space Operations Centre (ESOC).

The security part of this Analysis has been prepared with the following aspects in mind

- ESA's large decentralized ground facilities for operations,
- the multiple organizations/users involved in the operations and the developments of ground data systems, and
- the large heterogeneous network structure enabling access to (sensitive) data which does involve crossing organizational boundaries.

An IOI GS data objects classification is introduced to determine the extent of the necessary protection mechanisms. The proposal of security countermeasures is oriented towards the European "Information Technology Security Evaluation Criteria (ITSEC)" whose hierarchically organized requirements can be directly mapped to the security sensitivity classification.

1 SECURITY SITUATION OF THE IOI GS

The first consideration for the formulation of the security situation is the threats to which elements of the IOI GS are exposed. The threats are a function of the operational environment of the system and sensitivity of the data being processed in the various IOI GS facilities. The three basic security threats /ITSEC/ to which the facilities are exposed to are:

- the unintentional or unauthorized disclosure (loss of confidentiality),
- the unintentional or unauthorized modification (loss of integrity), and
- the unintentional or unauthorized destruction (loss of availability).

The causes of these threats are not only misuse.

Weak points in the Information Technology, human failure, and acts of God are also essential reasons.

The threats can be of differing significance to different types of data. For example, for a flight element, the disclosure of operations commands represents little or no threat, while their modification can cause serious damage, at worst the physical destruction of the spacecraft and the death of the crew if more than one command was modified.

The second consideration would be the assessment of the materialization of these threats in the context of the IOI GS environment. Assumed impacts could be estimated in terms of

- loss/injury of human life
- destruction of/damage to material and/or equipment
- degradation of functionality
- delay of mission

as for example laid down in the ESA standard ESA PSS-01-40.

The problem lies now in the creation of appropriate IOI GS units (function, data, or facility), which enable the association with some sort of security classification. The solution of this problem would result in a categorization scheme for the selected IOI GS units. Such a categorization scheme would simplify the identification of weak points and allow the assignment and evaluation of countermeasures to reduce the security risk.

2. APPROACH ADOPTED

The approach adopted in this analysis is illustrated in *Figure 1*. It describes basically the method of categorizing data into sensitivity levels which are applied to the IOI operations systems. This includes the definition of data objects (for example Payload Experiment Results, Telemetry, Platform Operations Commands) used within the IOI GS. These data objects are associated with their environment so that it is possible to imagine potential threats and their materialization.

The worst impact assessment leads to one of five consequence classes (see chapter 4) for each of the analyzed data objects. Thus it becomes apparent

which data needs to be protected and which not. The results of this assignment are summarized in decision tables (see chapter 6).

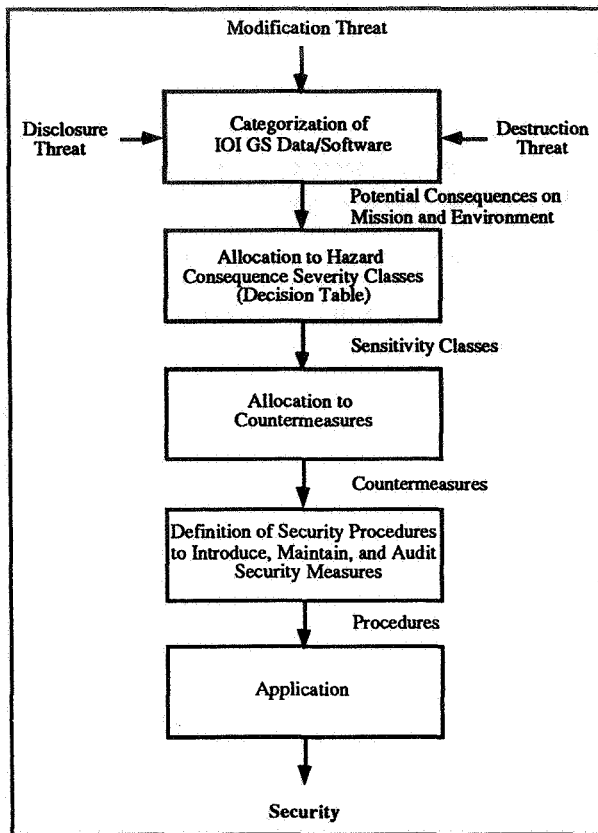


Figure 1: Approach Adopted

The sensitivity of the considered data object determines the effectiveness of the measure(s) to be implemented to counter the risk (see chapter 8 and 9).

3 SECURITY THREATS TO THE IOI GS

The security threats as pointed out above must be applied to all phases of the mission(s), including the preparation, the operation, and the maintenance. The following list of some of the most important threats is particularly related to the nature of IOI GS.

1. Introduction of unapproved codes during software development in terms of (non-) correctness and potential to be "infected". This is especially important due to the involvement of the large number of firms entrusted with the task of developing "ESA software".
2. As a result of the above mentioned point the access to confidential design documentation (hardcopies and on-line documents) by potential misusers is relatively easy.
3. The degree of decentralization of the Ground-Facilities-for-Operations (GFO) necessitates

complex communication links between the centers, as well as within the centers and connected sites themselves. This complexity (large number of nodes to be interconnected) increases the opportunities for potential intruders to "break in" (for example in order to infect system software or to obtain secret experiment data from a user).

4. The large number of information sources necessary to operate the IOI facilities increases the probability of incorrect decisions based on illegally modified or unintentionally falsified information.
5. The architecture of the IOI GS control centers allows access to sensitive data from a number of different work stations for each (authorized) user. This creates strong demands on the data protection mechanisms against erroneous and unintentional manipulation of data.
6. The use of state-of-the art, ergonomically optimized Human Computer Interfaces (HCI) may also be useful for people having gained access to a computer for malicious purposes. Then the "help functions" could be too helpful.
7. The freedom of the User Home Bases (universities, research centers) to run their on board experiments (payload control, processing of experiment data) and develop their own on board software for this purpose involves the danger that the user's software cannot be controlled in the same way as ESA software. That means secure, software controlled operations are difficult to achieve.
8. The repeated updates/upgrades/ replacements required to be carried out on IOI and GS systems (by ESA as well as by users) call for security considerations of the same kind as the original development.
9. The maintenance of ESA data processing facilities through "outside" suppliers holds the danger of ease access to hardware, software components and critical data (for example replacing of hard-disk, or software updates). This is especially important for re-arranging of system configuration or system adjustment. - If system changes can only be carried out by the supplier, a dependency in a very sensitive area arises.
10. For safety and availability reasons backup operations centers must be provided (for example having an additional stand-by control centers during critical operations). This further increases the security risk for data/software as well as for communications and people (see above).

4 CONSEQUENCE CLASSES

The first stage in the creation of security classes is the definition or application of some sort of consequence classes. Each piece of software or data can be associated with one of these consequences classes as

suming the worst impact caused by the materialization of a potential threat. Based on ESA PSS-01-40 (Hazard Consequence Severity Categories) five security related consequence classes have been defined. They comprise in short:

1. CAT (Catastrophic): Loss of life, life threatening or permanently, disabling injury.
2. CRT (Critical): Temporary non-availability of some flight systems or some of its vital parts that could cause the failure of experiments.

In both cases, the only recovery is to abort the current mission.

3. MAR (Marginal): Temporary non-availability of some systems.
4. SQM (Subsequent Mission): No consequence for the current mission. Unauthorized use of data could compromise potentially future mission(s).
5. NEG (Negligible): Minor problems, without serious impacts on the mission.

5 DATA OBJECTS

A further step to reach a security classification is the definition of data objects used within the IOI GS. Taking into account the risks involved in the consequence classes described above, these data objects provide the basis for the worst impact analysis.

The following list contains the generic data objects for those pieces of data defined for the IOI GS:

- Crew Data
 - Medical Data
 - Voice/Video Data
- Platform Data
 - Commands
 - Telemetry
- Payload Data
 - Commands
 - Telemetry
- Experiment Data
- Ground Segment Data
 - Command Process-Line
 - Telemetry
 - Engineering Support, Test and Training
 - Characteristics Data Base
 - Reports
 - Experiment Data
 - Filed Experiment Data
 - Network and GFO Management Data

6 DECISION TABLES

Each of the above mentioned data objects (and the software which handle it) was evaluated in respect of the threats of disclosure, destruction, and modification bearing in mind the particular security situation of the IOI GS described in chapter 3. The results were summarized in tables of which one example is shown below:

SENSITIVITY CLASS	CAT	CRT	MAR	SQM	NEG
GS DATA					
Experiment Data		mod/ des			dis
...					

Table 1: Decision Table (Example)

Abbreviations of the considered threats as used in the tables:

- mod = modification (covert and open)
- des = destruction (both irretrievable destruction and temporary inaccessibility)
- dis = disclosure

The decision tables assign a sensitivity class to each data object. The class is derived from their Hazard Consequence Severity Category. Thus in the example shown in Table 1 the "Experiment Data" is categorized as critical in the event of its modification and destruction (mod/des). The disclosure risk (dis) is negligible.

7 CRITICAL IOI GS DATA OBJECTS

The evaluation of the data objects has revealed that in the majority of cases the modification (covert and open) and destruction of data and software can have more serious implications on the mission and its environment than the disclosure risk. This finding should justify placing emphasis on the protection against the materialization of these kinds of risks.

Special attention must be paid to the very sensitive data objects, such as "PF operations commands", "system operational procedures", including their telemetry and the "network and GS management data". Their modification could result in catastrophic consequences. It is therefore important that these kinds of data are treated separately in relation to processing storage and transmission.

Another very sensitive type of data is that of "network and GFO management", since it contains all security implementation characteristics, such as compu-

ter hardware, software and network configurations or access control information. Hackers usually start by modifying this sort of data once they have "broken in". This provides them with the opportunity to continue with further (malicious) manipulations. So the administration of this data must provide a very effective mechanism to protect it against unauthorized disclosure.

In this context it is worthwhile noting that long term investigations carried out by security consultants dealing with a broad spectrum of customer establishments have found out that the most important threat of errors and omissions is caused by honest employees, who make mistakes in data entry, data update, changes in applications, etc.

8 COUNTERMEASURES

The main goal of the study is to describe a systematic method for identification of threats and the allocation of suitable countermeasures. The estimated degree of sensitivity of the threatened data object should determine the requirements on the systems to be designed.

A suitable means of achieving this goal is the use of the European IT Security Evaluation Criteria /ITSEC/ for two reasons:

1. They provide a basis for a standardized formulation of security aspects in the design documentation. That means Basic Security Functions as laid down in ITSEC:
 - Identification and Authentication,
 - Access Control,
 - Accountability,
 - Audit, and
 - Object Reuse
 are used accordingly to their ability to cover the actual threat(s) of each security relevant system to be analysed. In the same way the telecommunication security of all end-to-end telecommunication services can be examined by means of:
 - peer entity authentication,
 - access control,
 - data confidentiality,
 - data integrity,
 - data origin authentication, and
 - non-repudiation.
2. The security criteria give the sponsors valuable help for the formulation of requirements on and the Evaluation of Correctness and Effectiveness of the security characteristics. The ITSEC distinguish seven hierarchically structured Evaluation Levels. Each Evaluation Level defines assurance requirements with regard to the Correctness of
 - the Development Process (Requirements, Architectural Design, Detailed Design, Im-

- plementation),
 - the Development Environment (Configuration Control, Programming Language, Compilers, Developers Security),
 - the Operational Documentation (User Documentation, Administration Documentation), and
 - the Operational Environment (Delivery and Configuration, Start-up and Operation)
- and the Effectiveness of
- the Construction (Suitability of Functionality, Binding of Functionality, Strength of Mechanisms, and Construction Vulnerability) and
 - the Operation (Ease of Use, and Operational Vulnerability).

The following table shows a possible assignment of the ITSEC Evaluation Levels to the Sensitivity Classes:

Evaluation Level	Sensitivity Level
E1	NEG
E2	SQM
E3	MAR
E3	CRT
E4	CAT

Table 2: Assignment of Evaluation Levels

The Correctness and Effectiveness requirements on the systems are suited to the sensitivity of the data they process. That means the quality of a security related system increases with the sensitivity of its data. For example the ITSEC require from E2 onwards, the support by a configuration control system, or from E4 onwards, a formally specified model of security.

TECHNICAL COUNTERMEASURES	C A T	C R T	M A R	S Q M	N E G
System Software					
System Software should provide mechanisms for the handling of data of different sensitivity levels. It should contain the following basic security functions as integral parts:					
- identification and authentication	X	X	X	X	X
- access control	X	X	X	X	X
- accountability	X	X	X	X	
- audit	X	X	X	X	
- object reuse	X	X	X		

Table 3: Structure of Technical Countermeasures

TECHNICAL COUNTERMEASURES	C A T	C R T	M A R	S Q M	N E G
Communication					
Use of secure/certified network or communication software and hardware preferably as integral part of the underlying operating system. In particular containing means for:					
- peer entity authentication	X	X	X	X	X
- access control	X	X	X	X	X
- data confidentiality	X	X	X	X	
- data integrity	X	X	X	X	
- data origin authentication	X	X	X	X	
- non-repudiation	X	X	X		
Data Bases					
Database(s) which administer for example the network management data, engineering support data, and characteristics data must be considered as very critical. The following set of exemplary security functions should be implemented:					
- identification and authentication	X	X	X	X	X
- administration of user, roles (user with special attributes), and process rights	X	X	X	X	
- access only possible using specially established processes	X	X	X	X	
- defining new types, granting or revoking access rights to existing types only by authorized users, initiated by this users via a trusted path	X	X	X	X	
- access to objects of certain types only via fixed established processes	X	X	X	X	
- verification of user roles, and process rights	X	X	X	X	
- auditing definition or deletion of types	X	X	X	X	X
- auditing granting or revoking of access rights for objects or object types	X	X	X	X	
- auditing remote data base access	X	X	X	X	

Table 4: Structure of Technical Countermeasures

A possible assignment of the functional requirements show Table 3 and 4 for the areas of systems software (operating systems), data bases, and communications. The crosses in the right columns indicate me-

rely that the functional requirements in the left column have to be implemented for a certain sensitivity class. For the Correctness and the Effectiveness of these requirements the hierarchy of ITSEC can be used.

These examples have a high abstraction level and give guidance for structuring requirements rather than providing them. More detailed specifications can be formulated such as in the following table:

TECHNICAL COUNTERMEASURES	C A T	C R T	M A R	S Q M	N E G
Logical Access					
Automated audit procedure, allowing the supervision and control of resource usage. It should detect successful or attempted misuse of resources, software, and data. This requires the design of standardized audit interfaces to be implemented in each system (for example Workstations) to be monitored. Each system reports action to a centralized audit manager.	X	X	X		
...					
Measures against "hackers" and "crackers" for example:	X	X	X	X	
- avoiding naming of organization					
- avoiding identification of hardware					
- not using separate prompts for User Name, ID, and password					
- prohibiting message "Unauthorized Access ..."					
- protection of program code for example by:					
- submitting program to administration of access rights					
- frequent check of program size and creation date					
- not leaving sensitive data or software on the network if avoidable.					
...					

Table 5: Example of Security Requirements

9 ORGANIZATIONAL CONTROL

This chapter has been added to stress the fact that technical countermeasures alone, as raised in the pre

vious chapter, are insufficient to implement an effective security system.

A weak point is the users themselves: Many of them are unaware of the vulnerability of the data they operate with their systems. Highly sophisticated technical security measures can be useless, if users do not secure their working areas (for example simply by locking doors, log off the work stations from the network, or by mentally noting their passwords rather than keeping a written note on their desk).

Technical measures must therefore be complemented by organizational means, which control their implementation as well as ensuring their maintenance and development.

ORGANIZATIONAL CONTROL	C A T	C R T	M A R	S Q M	N E G
Establish independent security staff with authority to act full time for the entire organization. This staff should be responsible for the following issues:	X	X	X	X	X
Check (screening) and selection of personnel corresponding to the task they are or will be entrusted with. For example, only well educated and experienced staff for development/operation of critical software/data should be allowed. External personnel such as maintenance staff having access to data processing facilities and consequently to sensitive data as well as cleaning staff, plumbers, electricians (theft, espionage) must also be taken into account.	X	X			
...					
Programme to ensure security awareness of staff and users, including the requirement to sign a statement to acknowledge understanding of their responsibilities.	X	X	X	X	X
...					

Table 6: Complementing Organizational Measures

Table 6 represents a subset of lists of suggestions to be seen as counterparts to the tables containing the technical countermeasures. They serve as a possible input for the agency's security policy that in general constitutes the proceduralization of the technical se-

curity measures. Which one, or which set of the proposed procedures will finally be adopted in this security policy and the way of adopting them depends on the overall security architecture of the IOI GS.

From the organizational point of view data security is closely related to the Quality Management within an organization. That means, derived from a general security policy, procedures are implemented which ensure their introduction, maintenance, and ongoing development. These procedures are hierarchically structured in the same way as the technical countermeasures and reach to the project level. Their adherence and effectiveness must be audited in the same way as it is known from an effective Quality Assurance System.

10 CONCLUSION

The described procedure for the elaboration of a security concept can be applied to a lot of organisations which would certainly be worthwhile in many cases. Unfortunately security awareness arises only when something unexpected occurs (espionage, virus attack, failures, etc.).

The approach as described in this paper can prepare the ground for finding a way of improving security within an organization. The approach is based first of all on the definition of data objects, whose sensitivity determines the correctness and the effectiveness of the security enforcing functions and the extent to which organisational measures has to be complemented. In order to systemize the design procedure, proven standards can be applied.

This procedure only has a chance of success, if it is regarded as necessary by the management of an organisation and if the employees are motivated accordingly. Data security is an ongoing task and its efficiency depends considerably on how it is planned and supervised by the security staff.

11 REFERENCES

- GISA1 German Information Security Agency (GISA)
Bundesanzeiger ISBN 3-8 88784-192-1
Criteria for Evaluation of Trustworthiness of Information Technology (IT) Systems
1st Version, 1989
- ITSEC Information Technology Security Evaluation Criteria (ITSEC)
Harmonised Criteria of France - Germany - the Netherlands - the United Kingdom -
Version 1.2; Juni 1991

MISSION AND SCIENCE ACTIVITY SCHEDULING LANGUAGE

Larry G. Hull
Software and Automation Systems Branch, Code 522
NASA/Goddard Space Flight Center
Greenbelt Maryland 20771, USA

N94-23938

ABSTRACT

To support the distributed and complex operational scheduling required for future National Aeronautics and Space Administration (NASA) missions, a formal, textual language, the Scheduling Applications Interface Language (SAIL), has been developed. Increased geographic dispersion of investigators is leading to distributed mission and science activity planning, scheduling, and operations. SAIL is an innovation which supports the effective and efficient communication of scheduling information among physically dispersed applications in distributed scheduling environments.

SAIL offers a clear, concise, unambiguous expression of scheduling information in a readable, hardware independent format. The language concept, syntax, and semantics incorporate language features found useful during 5 years of research and prototyping with scheduling languages in physically distributed environments. SAIL allows concise specification of mission and science activity plans in a format which promotes repetition and reuse.

Key Words: Scheduling, language, data representation, distributed mission operations

1. DISTRIBUTED OPERATIONAL SCHEDULING

Operational scheduling for NASA missions is becoming more geographically distributed and more operationally complex (Ref. 1). Increased geographic dispersion of investigators (including Japanese and European) is leading to distributed mission and science activity planning, scheduling and operations. The growing number of sophisticated instruments that generate very high data rates and volumes is leading to a requirement for more complex and more automated planning, scheduling, and resource management.

The focus in this paper is on the scheduling of NASA mission operations, specifically on the scheduling of spacecraft resources to support spacecraft operations and geographically dispersed investigators. Figure 1 is an end-to-end scheduling environment (Ref. 2).

From the point of view of the network, scheduling serves to allocate shared communications resources. From the point of view of the mission, scheduling serves to maintain the health and safety of the spacecraft and to allocate shared spacecraft resources.

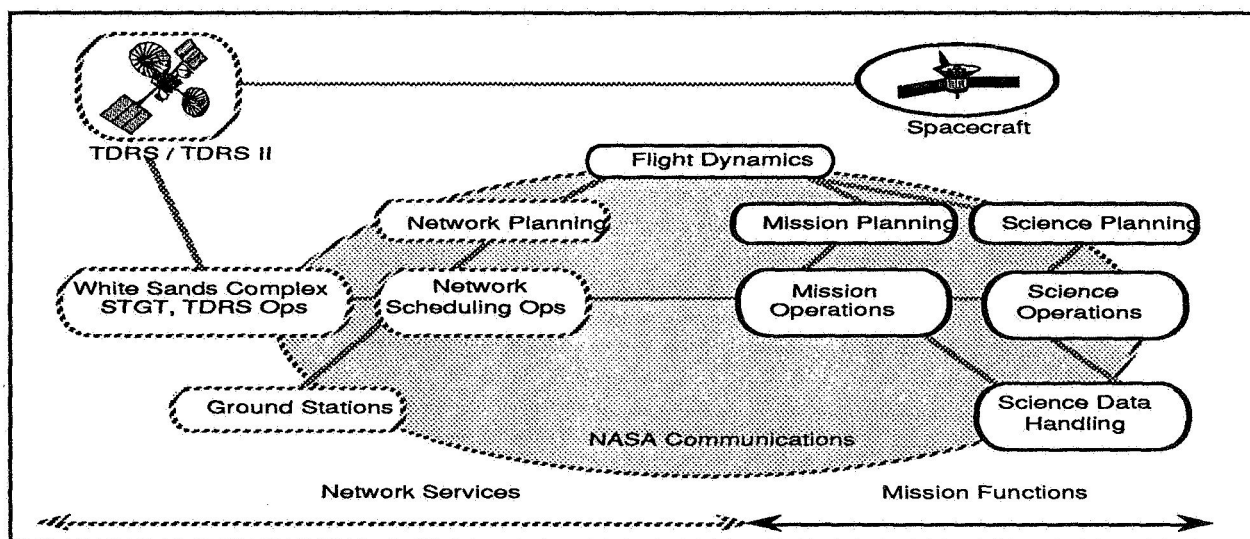


Figure 1. End-to-end Scheduling Environment

From the point of view of investigators, scheduling serves to maintain the health and safety of the science instruments and to support science data acquisition. Although all use the same or similar data for generating schedules which must eventually be consistent, objectives and techniques differ.

2. SCHEDULING DATA REPRESENTATION

As indicated, the scheduling of NASA mission operations is becoming an increasingly intricate and distributed process. To support future mission operations, scheduling information must be communicated effectively among physically dispersed applications in distributed scheduling environments. The scheduling information representation must allow expression of complex, variable scheduling requirements. For effective and efficient scheduling, the representation must also provide sufficient flexibility to accommodate changing requirements and scheduling alternatives. Finally, the representation should be concise, unambiguous, and in a hardware-independent format that can be processed by computer.

The Software and Automation Systems Branch at GSFC has addressed the problem of defining a scheduling data representation for two specific scheduling domains: communications support scheduling and spacecraft resource scheduling (Ref. 3). Each domain is a physically distributed scheduling environment with several different scheduling applications running on several different hardware configurations. Consequently, a suitable data representation could not presume a specific physical topology, a specific communications protocol, or a specific operations scenario. On the other hand, efficient and effective scheduling required reusable scheduling data representations supporting repetitive activities, sufficient flexibility to express often complex scheduling requirements and constraints, and the capability to communicate scheduling alternatives. The scheduling data representation had to be self-documenting, easily maintained, and physically and logically portable. Requirements analysis (Ref. 4) concluded that a formal, textual language was needed.

3. SAIL

Requirements analysis led to the definition of the Scheduling Applications Interface Language (SAIL).

SAIL is a formal, textual language (Ref. 5) for expressing scheduling information communicated between nodes of a distributed, activity scheduling system. SAIL provides interoperability between different scheduling applications running on different hardware configurations. SAIL expresses scheduling information in a clear, concise, unambiguous format that can be processed by computer.

SAIL expresses the inputs to the scheduling process, the outputs of the scheduling process, and other transactions needed to maintain schedules in an operational environment. SAIL expresses nominal and contingency operations plans for instruments and spacecraft, the constituent activities of the operations plans, the resource needs of these activities, and the constraints that restrict when these activities can be scheduled. SAIL also expresses both the resources available to satisfy activity resource needs and the resources allocated to activities (the schedule).

SAIL is intended to be used in the repetitive scheduling process that occurs throughout the life of a mission. In a typical operational environment, users:

- develop plans that reflect high level goals and objectives for an instrument or facility,
- derive scheduling information from these plans, e.g., instrument activities and target or view times,
- transmit scheduling information and requests to a scheduling facility,
- schedule local resources in accordance with instrument schedules received from scheduling facilities, and
- transmit rescheduling information as needed.

Scheduling facilities typically will:

- distribute appropriate scheduling information to users,
- receive and store user-generated scheduling information and requests,
- coordinate with resource providers to build integrated schedules,
- disseminate schedules to users, and
- accept new inputs and reschedule to meet changing science needs.

4. A SCHEDULING NODE

Figure 2 shows a functional architecture for a scheduling node that accepts scheduling requests from multiple users. The scheduling node in the

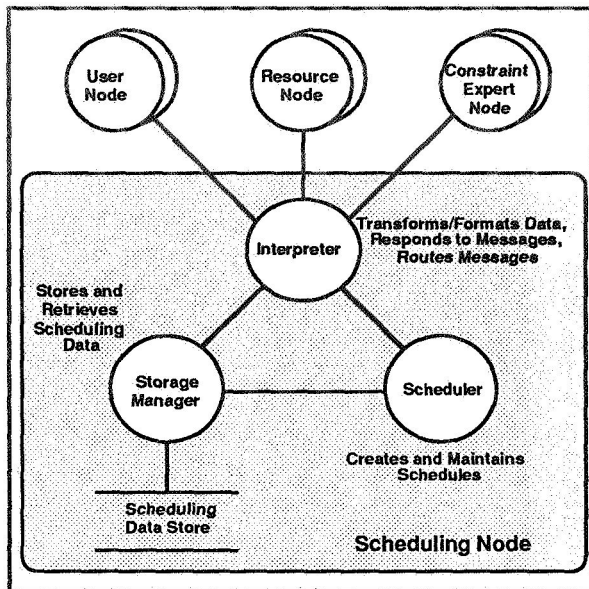


Figure 2. Scheduling Node Functional Architecture

figure is somewhat representative of other nodes in the network, since many applications in a scheduling domain perform some scheduling functions (e.g., user nodes schedule local resources.)

The SAIL interpreter accepts messages and interacts with the scheduler and the storage manager. The interpreter transforms the incoming data for the scheduler and the storage manager. Interpreting a message may simply cause the receiving node to send back an acknowledgment message. In many cases, the message may have more substantial effects. A message that specifies a request for resources (an activity) for a future schedule results in this request being added to the data store at the receiving node. A message that requests a change be made to a current schedule results in a reallocation of resources.

The scheduler creates and maintains schedules, including the current on-line schedule. This function is typically performed by operations personnel with assistance from automated scheduling software. The scheduler interacts with the SAIL interpreter and processes messages that affect the current on-line schedule. The scheduler interacts with the storage manager to store schedules and to retrieve information needed to create and maintain schedules.

The storage manager services requests from the SAIL interpreter and the scheduler to store, update, or access information in its data store. The data store contains the information needed to create schedules and the schedules that result from the scheduling

process. The types of data stored typically include requests for resources, resource availability information, constraint information, and schedules.

5. DATA MANAGEMENT

The scheduling problem is a data management problem. A scheduling node typically maintains several schedules. Each of these schedules consists of many pieces of data (requests for resources, constraint information, resource availabilities, etc.). Some data is relevant over a long period of time; other data is relevant for only one or a few schedules.

SAIL contains data structures that make it suitable for complex, multi-schedule environments. SAIL statements enable users to specify the schedule(s) for which the data is valid. Since multiple schedules can be specified, data that is used repeatedly does not have to be retransmitted.

SAIL handles the Flexible Scheduling Request Concept. This concept is an alternative to traditional methods of fixed format scheduling. With fixed format scheduling, users submit requests that are either approved or disapproved. Users frequently submit many more requests than may be necessary in anticipation that some of the requests will be rejected. With flexible scheduling, users specify single or repeated activities, and enough flexibility in activity execution time and resource requirements to enable the scheduler to more optimally handle the request.

For instance, an activity might specify that the duration of an operation must be "exactly 2 hours" or that the duration must be "longer than 20 minutes." Flexible requirements can potentially reduce the number of conflicts in schedules, and may enable a scheduler to produce a more optimal schedule. For example, suppose Activity A has been scheduled at 2:00 p.m. on Tuesday but its requirements specify that it really can be scheduled any time during the day on Tuesday. Next, suppose that the scheduler receives a request to schedule Activity B only at 2:00 p.m. on Tuesday. Given the flexibility specified in the requirements of Activity A, the scheduler is free to move Activity A to another time in order to accommodate Activity B. The resulting schedule is therefore more optimal.

Figure 3 shows the flexibility options SAIL is capable of representing. The vertical axis illustrates repeatability--in the upper left corner, more activities

Increasing Repeatability	Schedule an activity each day starting at exactly noon	Schedule an activity sometime during each day
	Duration: 20 minutes Repetition: Every day Resource: IMAGER1 Start Time: 12:00 noon Period: Next two weeks	Duration: 15-20 minutes Repetition: Every day Resource: Any IMAGER Start Time: After 12:00 noon Period: Next two weeks
	Schedule one activity starting at exactly 15:12:36	Schedule one activity starting sometime after 12:00 noon
	Duration: 20 minutes Repetition: Once Resource: IMAGER1 Start Time: 15:12:36 Period: Tuesday	Duration: 15-20 minutes Repetition: Once Resource: Any IMAGER Start Time: After 12:00 noon Period: Tuesday
Increasing Flexibility		

Figure 3. Flexible Scheduling Request Concept

will be scheduled per request. The horizontal axis illustrates flexibility--in the lower right corner, more flexibility is specified in a request providing the scheduler with more scheduling options. In the upper right corner, the request is both repeatable and flexible--many activities can be scheduled from a request with a lot of flexibility giving the scheduler many scheduling options.

Figure 4 illustrates possible scheduling time windows, selected by a scheduler, based on a flexible request. The request includes the constraints that must be satisfied when scheduling a viewing activity but does not specify specific start and end times. The scheduler is given the option of determining when to schedule the activity, taking into consideration the set of constraints for this activity. In the example, for the set of constraints specified, the scheduler has the flexibility to schedule the viewing activity any time during the time windows shown as shaded boxes.

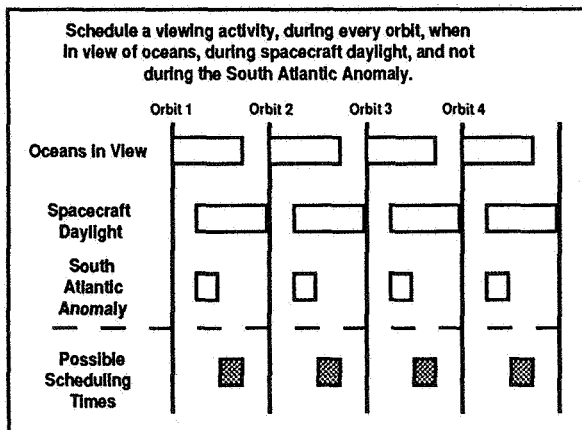


Figure 4. Possible Scheduling Time Windows

6. AN INTRODUCTION TO SAIL

6.1 SAIL Messages

Nodes in a distributed environment communicate through SAIL messages. There are two types of messages in SAIL: acknowledgment messages and declarative messages. An acknowledgment message acknowledges the receipt of a declarative message. Declarative messages contain planning and scheduling information and are used to transmit this information from one node to another.

A declarative message contains a header and a sequence of statements. The declarative message header specifies such information as the sender of the message, the intended receiver, and the time the message was sent. Each statement within a declarative message specifies an action and some data. The statement can be any of the following:

- a request for the receiver of the message to perform the stated action on the specified data,
- a response that indicates the success or failure of a request, or
- a notification that the sender of the message has already performed an action on the specified data (no response action is required).

A declarative message may contain any number of request, response, and notification statements in any order.

6.2 SAIL Statements

SAIL has four request actions, four response actions, and four notification actions. Each of these actions, together with a SAIL structure (defined below), form a SAIL statement.

6.2.1 Request Statement

Request statements are the "action" statements in SAIL. Through request statements, the sender can cause changes at a receiver's (remote) node. A user node sends messages containing request statements to a scheduling node to accomplish each of the following:

- add an activity to a schedule,
- change an activity already on a schedule, and
- delete a scheduled activity from a schedule.

Request actions are CREATE, REPLACE,

DELETE, and DESCRIBE. The first three are data management actions that create, update, or remove SAIL structures located at another node. The fourth request action, DESCRIBE, allows a node to obtain the definition of a structure stored at another node.

6.2.2 Response Statement

Response statements consist of the reserved word RESPONSE_TO followed by the request action and an indication of the success or failure of the request. Response actions are the same as request actions.

6.2.3 Notification Statement

Notification statements consist of a notification action usually followed by a structure. Notification actions are: CREATED, REPLACED, DELETED, and DESCRIBED. Notification statements are used to notify other nodes that a change has occurred. Notification statements differ from response statements in that a notification statement is not the result of any request statement. For instance, if a scheduling node reschedules some events in response to a change in predicted resource availabilities, it notifies the affected users that a change has occurred.

6.3 SAIL Structures

Efficient planning and scheduling in a distributed environment requires the capability to express several types of information. For instance, user nodes must be able to communicate their resource needs to the provider of the resource or to the control center responsible for allocating the resource. Scheduling results (resource allocations) must be returned to the user nodes that requested the resources.

This information is contained in the SAIL declarative message. SAIL request and notification statements in a SAIL declarative message identify an action and a structure that is the object of the action. Structures express an input or output of the activity scheduling process. There are five structures defined in SAIL: activities, plans, allocations, profiles, and meta-schedules. Therefore, a facility may request another facility to create, describe, delete, or replace an activity, plan, allocation, profile, or meta-schedule.

6.3.1 Activities

Activities specify the requirements of spacecraft and instrument operations. These requirements consist of

resource needs and other restrictions that limit the start time, end time, and duration of the activities. A specification includes the name of the activity, the priority of the activity with respect to other activities requested by the same node, any requirements to coordinate this activity with other activities and spacecraft events, and options that a scheduler should try in case the activity cannot be scheduled as initially specified. Requirements can be specific or flexible, as previously described.

Schedulers use activities in determining how to allocate available resources. When an activity is "scheduled," resources are allocated for the operation described by the activity. Resource allocations are expressed using the allocation structure.

6.3.2 Plans

The plans provided in SAIL represent a sequence of activities that some facility or system performs. SAIL plans are expressed as repeating sequences of activity structures and are often reusable. For instance, the plan "perform a tape recorder dump every fourth orbit using TDRS-East" is a reusable SAIL plan.

6.3.3 Allocations

Allocations represent the output of the scheduling process. An allocation expresses assignment of resources needed to perform an operation or a part of an operation with specific start and end times for each resource assignment. Collectively, the allocations for a given time period (e.g., a day) represent a schedule.

6.3.4 Profiles

Profiles are a general data representation structure primarily used to represent numeric data that varies over time. The data in profiles consist of words, numbers, and character strings; thus, profiles can express a variety of types of information, such as:

- the varying amount of a resource available over time as activities obligate and release the resource,
- user antenna view (UAV) times, and
- user spacecraft ascending node, descending node, apogee, and perigee.

Profiles that represent resource amounts, such as available power, often constrain activities. For instance, an activity might require 20 watts of power

for 30 minutes. The power profile specifies the available power and restricts the scheduling of the activity. Figure 5 shows an example of an available power profile with a maximum value of 30 watts and a minimum value of 10 watts. The profile is defined for a 6-hour period. Using this available power profile, an activity that requires 20 watts of power could be scheduled only from 6:30 to 7:30.

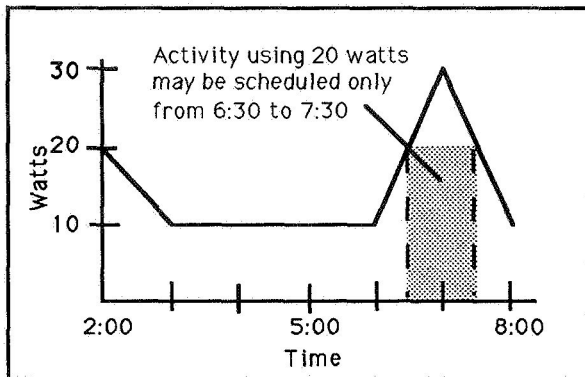


Figure 5. Power Profile

6.3.5 Meta-schedules

A SAIL meta-schedule structure contains a schedule (i.e., a list of allocations) and additional information about the schedule. This information consists of the inputs to the scheduling process (SAIL activity, plan, and profile structures) and metrics that express the quality of the schedule with respect to scheduling goals. A meta-schedule structure provides a record of the information used to create a particular schedule and a record of the results of the scheduling process. Structure headers specify the names of the schedules (more specifically, meta-schedules) to be used when performing an action on a structure.

SAIL meta-schedules are either active or inactive. An inactive meta-schedule does not contain specific assignments of resources (i.e., it does not contain allocations). Meta-schedules are initially inactive. During this period, a scheduling node that creates schedules collects requests for resources. At some point in time, the scheduling node "creates a schedule." To "create a schedule" is to make an active meta-schedule from an inactive meta-schedule (i.e., to assign resources to activities and create the allocation structures that specify the assignments).

7. SUMMARY

SAIL is a formal, textual language developed to

support effective and efficient communication of scheduling information among physically dispersed applications in distributed scheduling environments. The language is an innovation which allows both for expression of complex, variable requirements and for concise specification of mission and science activity plans in a format which promotes repetition and reuse. SAIL's ability to express plan flexibilities and alternatives reduces the need for schedule iterations. SAIL does not presume a specific physical topology, communications protocol, or operations scenario.

Experience with predecessor languages in prototyping environments indicates that SAIL can express the control and data information that must be communicated between nodes in a distributed scheduling environment, and the complex scheduling requirements of instruments and spacecraft. SAIL is viewed as a relatively high-payoff, low-risk approach for operational implementation.

8. REFERENCES

1. Hull, Larry G. et al. 1990. Distributed Planning and Scheduling for Instrument and Platform Operations. In AIAA/NASA Second International Symposium on Space Information Systems. Washington, DC: American Institute of Aeronautics and Astronautics.
2. Moe, Karen. 1992. End-to-End Planning and Scheduling Systems Technology for Space Operations. In 43rd Congress of the International Astronautical Federation. Paris: International Astronautical Federation.
3. Buford, Carolyn. 1992. Scheduling Data Representation: Concept and Experience in Code 520, (DSTL-92-010). Greenbelt, MD: NASA/Goddard Space Flight Center.
4. Zoch, David. 1991. Requirements for the Scheduling Applications Interface Language (SAIL). Seabrook, MD: Loral AeroSys.
5. Sary, Charisse. 1992. The Scheduling Application Interface Language (SAIL) Reference Manual, (DSTL-91-021) Revision 1. Greenbelt, MD: NASA/Goddard Space Flight Center.

SPECIFYING DESIGN CONSERVATISM: WORST CASE VERSUS PROBABILISTIC ANALYSIS

Ralph F. Miles, Jr.

N 9 4 - 2 3 9 3 9

Jet Propulsion Laboratory
Pasadena, California, USA

ABSTRACT

Design conservatism is the difference between specified and required performance, and is introduced when uncertainty is present. The classical approach of worst-case analysis for specifying design conservatism is presented, along with the modern approach of probabilistic analysis. The appropriate degree of design conservatism is a tradeoff between the required resources and the probability and consequences of a failure. A probabilistic analysis properly models this tradeoff, while a worst-case analysis reveals nothing about the probability of failure, and can significantly overstate the consequences of failure. Two aerospace examples will be presented that illustrate problems that can arise with a worst-case analysis.

Key Words: Conservatism, design, analysis, worst-case, probability.

1. INTRODUCTION

Design conservatism is the difference between specified and required performance, and is introduced when uncertainty is present. It can appear in several forms--as a margin added to the performance requirements, or as an overstatement in the associated risk of failure, either in the consequences or the probability of failure. In any form, it forces more resources into some aspect of the design.

In practice, design conservatism is specified and implemented by a wide range of people, from top managers to technicians. In protecting against failures, the designer must ask:

- What can go wrong?
- How likely is it?
- What are the consequences?
- What can be done to reduce the likelihood?
- What can be done to mitigate the consequences?

It is out of this questioning process that candidates for design conservatism arise. Where the consequences of a power outage are severe, facilities supply their own backup power. Uncertainty exists as to how long the power would be out, and how much power would be needed during this period. The design conservatism would be the overcapacity added to the backup power source.

An out-of-control nuclear power plant can go super-critical with a subsequent catastrophic emission of radioactive material. While this has not happened in the U.S., nevertheless Chernobyl makes this a credible scenario. The consequences are sickness, loss of life, and the environmental damage and cleanup costs. Conservatism could be implemented by overstating the likelihood of an accident and the severity of the consequences, and designing to meet environmental requirements consistent with the overstated risk.

2. WORST-CASE ANALYSIS

Worst-case analysis for a design is a technique used to resolve uncertainties, and is one way to implement design conservatism. A design margin is added to bring the minimum performance level of the design up to the maximum performance level that could be required. An additional design margin may be added to account for unknown effects not included in the worst-case estimates. In principle, worst-case analyses negate any requirement to consider uncertainty.

A worst-case analysis for specifying a design may be appropriate if the following conditions can be met:

- (1) The worst-case scenario for the analysis is truly a worst case.
- (2) The adverse consequence associated with the worst-case scenario is acceptable.
- (3) The design consistent with the worst-case analysis is feasible.

For example, if it were not possible for a bird weighing more than 12 pounds to impact an aircraft windshield at more than 600 miles per hour, then a design that could be guaranteed to survive this impact would be consistent with a worst-case analysis. If it were merely improbable that the bird would weigh more, or improbable that the plane would be traveling at a higher speed, then the design would not truly correspond to a worst-case analysis.

With respect to Condition (1), stating that it is not credible that the bird would weigh more, or that it is not credible that the plane would be traveling at a higher speed, is not sufficient. The touted virtue of a worst-case analysis is that you don't have to specify the uncertainty. A statement that a scenario is

not credible only says that the scenario has a very small probability of occurrence, not that the scenario is impossible. Thus the worst-case analysis implicitly becomes a probabilistic analysis. Similarly, with respect to Condition (2), the risk associated with the worst-case analysis must be acceptable because the adverse consequence associated with the worst-case analysis is acceptable, not because the probability of occurrence is acceptably small.

What's wrong with designs consistent with worst-case analyses? Nothing, if you can satisfy Conditions (1) and (2), retain feasibility, and spare the additional resources. But this rarely happens in practice. What often is claimed to be a design consistent with a worst-case analysis is more often a design by fiat--the design will perform under stated conditions which may or may not be exceeded--or is a design consistent with an analysis for all credible conditions masking as a worst-case analysis. For example, worst-case analyses for electronic equipment assume that the environment and the power source will be within specification.

Worst-case analyses are most appropriate at the lowest level of systems, such as specifications for electronic parts and circuits, where a detailed analysis of the environment for each individual part would be impractical and unnecessary (Ref. 1). At higher levels in a system, designs consistent with worst-case analyses impact limited resources.

All designs exist in a resource-constrained environment. Thus, exceeding the requirements of one aspect of the design inevitably comes at the expense of something else. The increase in the strength of a structural element comes at the expense of weight--thus reducing payload. A triply-redundant design costs more than simple redundancy.

3. MEASURING PROBABILITY

It is a maxim of good management that in order to manage something you must be able to measure it (Ref. 2). To manage uncertainty and subsequently apply conservatism to a probabilistic design, you must be able to measure the risk--probability and consequences--associated with the uncertainty. While there are undeniable problems with measuring consequences, the greatest controversy arises in the measurement of probability.

Can probabilities be measured--or do they even exist--for events where the probability of occurrence is so low that few if any events have ever been observed? The answer to this question comes from theories of probability. There is little disagreement on how to use probabilities once they have been assigned. Nearly all analyses claiming to be probabilistic use the three Kolmogorov axioms (Ref. 3): (1) Probabilities are numbers equal to or greater than 0, (2) the probability of the universal event (sure thing--something happens) is 1, and (3) probabilities of mutually exclusive events add.

Simple arguments justify using Kolmogorov axioms for the long-run ratio of successes to trials, and probability is defined this way in many probability texts (Ref. 4). This ratio lies between 0 and 1, and these ratios add for mutually exclusive events.

Where few or no events have been observed, probabilities must be based on degrees of belief. One might believe based on some evidence that the probability of rain today is 2/3. Consistency with the Kolmogorov axioms, for example, requires that the probability of no-rain be 1/3. The most commonly used justification for the use of the Kolmogorov axioms for degrees of belief

comes from game theory (Refs. 5-7). It can be demonstrated that if you violate the Kolmogorov axioms with respect to your degree of belief about an uncertain event, it is possible for a broker to place a series of bets against you such that you will always lose, independent of the outcome of the event. Contrariwise, if you are consistent with the Kolmogorov axioms, such a series of bets is not possible.

4. PROBABILISTIC ANALYSIS

Apostolakis, in his *Science* article, "The Concept of Probability in Safety Assessments of Technological Systems," lists four steps to doing a probabilistic analysis for a system design (Ref. 8):

- (1) Structure the problem.
- (2) Quantify uncertainties.
- (3) Quantify preferences.
- (4) Choose among alternatives.

A probabilistic analysis proceeds by constructing a mathematical model representing the alternatives, the uncertain parameters, the consequences, and their interrelationships, and a model representing preferences for consequence attributes. Uncertainties are represented by random variables with associated probability distributions assessed by technical experts.

Preferences for attributes of the consequences are quantified by means of a value function, and it is in the value function that preferences for conservatism in the design should be expressed, e.g., the probability shall be less than one in a million that the design fails, or that one attribute of the consequences exceeds a stated amount. Finally, the alternative is chosen that maximizes the value.

Probability assessment techniques are discussed in the literature (Ref. 9), and have been applied in the nuclear power industry (Ref. 10 and 11). For a book exhaustive in techniques for modeling uncertainty, see Martz and Waller's *Bayesian Reliability Analysis* (Ref. 12). For introductions to probability assessment, see Merkhofer (Ref. 13) and Apostolakis (Ref. 14).

The product of a probabilistic analysis should include a statement of the central tendencies and spreads of the uncertain quantities. The central tendencies may be expressed as means or modes, and the spreads as standard deviations or fractile ranges. Even more desirable would be complete probability distributions, which can be derived from analysis or from Monte Carlo simulations.

The analysis should include all uncertainties, not just those easily measurable. It should incorporate, in order of increasing difficulty:

- (1) Uncertainties in the model parameters.
- (2) Uncertainties in the model selection.
- (3) Uncertainties in the decision process.

Unfortunately, too often design analyses are called "probabilistic" when only the uncertainties in the parameters of the model are addressed. Rarely are uncertainties in the model selection treated probabilistically, and almost never uncertainties in the decision process.

5. APOLLO RELIABILITY ASSESSMENT

President Kennedy's pronounced goal of getting a man to the moon and returning him safely before the end of the decade of the sixties initiated the Apollo Project. At times the goal seemed unattainable. Problems and uncertainties abounded--solar-particle radia-

tion and meteoroid hazards, an unknown lunar surface, and issues in designing and testing the Saturn V launch vehicle. The Apollo fire on the launch pad highlighted the difficulty of this project.

A lesser-known problem involved assessing the reliability of the Apollo Mission. George Mueller, the NASA Associate Administrator for Manned Space Flight during this period, says that an attempt was made to estimate the probability of mission success using bottom-up, piece-part failure rates and single-point-failure analyses (Ref. 15). When the original estimates for all the subsystems were combined, the probability of a successful lunar landing was about 5% (Ref. 16). Mueller ordered that a top-down estimate be made by looking at the success probability of each major event. The revised number came out to be 90% for the mission and 99% for getting the crew back, a success probability acceptably high.

The subsequent success rate for crews launched and safely returned has shown that the extreme pessimism of the bottom-up analysis was unwarranted. Whether this pessimism was the result of conservative estimates or true beliefs can't be known. Given the limited experience with space vehicles at that time, it seems reasonable that the engineers produced overly-conservative numbers to cover the uncertainties. As the senior manager, Mueller recognized the implications to the project if it really were true that the missions were likely to fail. Here conservatism as applied by the engineers didn't just yield a suboptimal design, it rendered the project infeasible.

This incident has had a lasting effect to this day on how NASA views numerical estimates of reliability. Quoting from a recent NASA Management Instruction (Ref. 17):

"It is crucial that when quantitative risk assessment methods are used, care must be exercised to ensure that all uncertainties are properly included in the analysis, and clearly displayed to avoid misinterpretation and misuse of the results."

6. THE GALILEO EXPERIENCE

The Galileo spacecraft was launched on October 18, 1989 on a 6-year flight to Jupiter, where it will release a probe into its atmosphere, and then go into orbit around the planet. As Jupiter is too far from the Sun for solar panels to be effective, plutonium-fueled radioisotope thermoelectric generators provide spacecraft power.

The National Environmental Policy Act (NEPA) and Presidential Directive/National Security Council PD/NSC-25 require that all launches with nuclear payloads undergo an extensive environmental impact analysis and review process. NASA, as the implementing agency, produced an Environmental Impact Statement (EIS) and the Department of Energy (DOE) issued a Safety Analysis Report (SAR). The Interagency Nuclear Safety Review Panel (INSRP), an independent review group, produced a Safety Evaluation Report (SER).

The conclusions of these reports differed in some respects. These differences were used in support of a motion for preliminary injunction against the launch of Galileo by litigating parties (Ref. 18):

"The SER's risk assessments are far more pessimistic than those in the Final EIS. In several instances, risk and radiological effects are 10 to 20 times greater than those in the Final EIS. . . . When, however, the estimates of risk in the SER proved to be higher, by magnitudes, than

in the . . . SAR . . . NASA simply rejected the conclusions of the SER . . . these actions violated NEPA . . ."

The Court subsequently denied the motion (Ref. 19):

"While the SER is slightly more pessimistic in its risk analysis, there is no evidence that the difference is statistically significant since both documents indicate that the risk is small. . . . The motion for a temporary restraining order is denied."

Were the differences between the SER, the EIS, and the SAR due to basic differences in engineering judgment, or were they due to conservative assumptions? Subsequent to the release of the SAR, the DOE issued a supplement to the SAR stating (Ref. 20):

"The source terms (plutonium releases) have been analyzed anew but with conservatism removed where it was judged to be excessive."

Similarly, for INSRP's SER (Ref. 21):

(Accident response and source terms incorporate) "reasonable conservatism . . . where understanding or physical descriptions were seriously lacking. . . . the source terms calculated tend to lean towards an upper bound of expectation." . . . (The reentry analysis) "is considered conservative."

While DOE's SAR and INSRP's SER differ, it is not possible to know the source of the differences. The conservative assumptions of both reports masked any differences that may have been due to engineering judgment.

In this and the preceding example, unwarranted conservatism was present. Further-

more, it was introduced in the analysis, which is the wrong place. We all want conservatism in designs--in the buildings we occupy, the planes we fly, and the medicine we take. What we shouldn't want is conservatism in the analysis. In the words of Zeckhauser and Viscusi, it amounts "to lying to ourselves about what we expect" (Ref. 22).

7. CONCLUSION

Imagine that you are the manager for a system undergoing a design review. One of your technical experts says he has done a worst-case analysis of a design that protects against all contingencies. Ask yourself these questions:

- Is the analysis truly worst-case?
- If it is a worst-case analysis, can I afford the design consistent with the analysis?
- If it is not a worst-case analysis, then how did the technical expert specify the most-unfavorable case for the analysis?
- What conservatism would I pick for the design?
- What would be the design if my conservatism were specified?

Design conservatism, properly applied, is necessary in that it protects against failure. This conservatism should appear as a specification for the design, and not a condition on the analysis. The degree of conservatism should relate to the probability and severity of the adverse consequences of a failure and should be traded off against the required resources. A probabilistic analysis properly models this tradeoff, while a worst-case analysis reveals nothing about the probability of failure, and can significantly overstate the consequences of failure.

8. REFERENCES

1. Design and Evaluation Inc. 1991. Worst Case Circuit Analysis Training Course. Laurel Springs, New Jersey. Third edition.
2. Daniel S. Goldin. 5 December 1991. Total Quality Management--A Culture Change. *TQM Lessons Learned*. Goddard Space Flight Center.
3. Kolmogorov, A. N. 1956. *Foundations of the Theory of Probability*. New York: Chelsea Publishing Co. Translation of original 1933 German edition.
4. Papoulis, Athanasios. 1984. The Meaning of Probability. *Probability, Random Variables, and Stochastic Processes*. New York: McGraw-Hill Book Company, I, Second Edition.
5. Earman, John. 1992. *Bayes or Bust? A Critical Examination of Bayesian Confirmation Theory*. Cambridge, Massachusetts: The MIT Press.
6. Kemeny, J. G. 1955. Fair betting and inductive probabilities. In *Journal of Symbolic Logic*, 20:263-273.
7. Shimony, A. 1955. Coherence and the axiom of confirmation. In *Journal of Symbolic Logic*, 20: 1-28.
8. George Apostolakis. 7 December 1990. The Concept of Probability in Safety Assessments of Technological Systems. In *Science*, 250: 1359-1364.
9. M. Granger Morgan and Max Henrion. 1990. Human Judgment About and With Uncertainty. In *Uncertainty: A Guide to Dealing with Uncertainty in Quantitative Risk and Policy Analysis*. New York: Cambridge University Press, 6.

10. J. W. Hickman, et. al. January 1983. *PRA Procedures Guide: A Guide to the Performance of Probabilistic Risk Assessment*. NUREG/CR-2300, I and II. U.S. Nuclear Regulatory Commission.
11. U.S. Nuclear Regulatory Commission. September 1984. *Probabilistic Risk Assessment (PRA) Reference Document. Final Report*. NUREG-1050-F. Washington, DC.
12. Harry F. Martz and Ray A. Waller. 1982. *Bayesian Reliability Analysis*. New York: John Wiley.
13. Miley W. Merkhofer. 1987. Criticisms and Limitations of Decision-Aiding Approaches. In *Decision Science and Social Risk Management*. Dordrecht, Holland: D. Reidel Publishing Company, 3.
14. George Apostolakis. 1988. Expert Judgment in Probabilistic Safety Assessment. In *Accelerated Life Testing and Experts' Opinions in Reliability*. Edited by C. A. Clarotti and D. V. Lindley. Elsevier Science Publishing Company Inc.: 116-131.
15. George Mueller. 1973. The Apollo Mission. In *Systems Concepts: Lectures on Contemporary Approaches to Systems*. Edited by Ralph F. Miles, Jr. New York: John Wiley.
16. Ben Buchbinder. March 19, 1992. "It is common knowledge at NASA Headquarters that the number was about 5%, but the study may never have been documented." Washington, D.C. NASA Office of Safety and Mission Quality. Phone conversation. A subsequent literature search failed to identify such a study. The number of 50% in Ref. 15 probably was a misquote.
17. NASA QS/Safety Division. 3 February 1988. *Risk Management Policy for Manned Flight Programs*. NASA Management Instruction NMI 8070. Washington, DC. National Aeronautics and Space Administration.
18. Florida Coalition for Peace and Justice, Foundation on Economic Trends, and Christic Institute vs. George Herbert Walker Bush, et. al. 4 October 1989. Memorandum of Points and Authorities in Support of Motion for Preliminary Injunction. United States District Court for the District of Columbia, Civil Action No. 89-2682-OG, 13.
19. Judge Oliver Gasch. 10 October 1989. Memorandum Concerning Civil Action No. 89-2682-OG. United States District Court for the District of Columbia: 11, 12, and 17.
20. General Electric Astro-Space Division. August 1989. *Supplement to the Final Safety Analysis Report for the Galileo Mission*. Philadelphia, Pennsylvania, Document No. 89SDS4221: 43.
21. Interagency Nuclear Safety Review Panel. September 1989. *Safety Evaluation Report for Galileo*, Document INSRP 89-01, I: VI-19.
22. Richard J. Zeckhauser and W. Kip Viscusi. 4 May 1990. Risk Within Reason. *Science*, 248: 559-564.

SIMSAT : AN OBJECT ORIENTED ARCHITECTURE FOR REAL-TIME SATELLITE SIMULATION

Adam P. Williams

N94-23940

Space Division
Cray Systems
Transome House
Victoria Street
Bristol BS1 6AH
United Kingdom

ABSTRACT

Real-time satellite simulators are vital tools in the support of satellite missions. They are used in the testing of ground control systems, the training of operators, the validation of operational procedures and the development of contingency plans.

The simulators must provide high-fidelity modelling of the satellite, which requires detailed system information, much of which is not available until relatively near launch.

The short timescales and resulting high productivity required of such simulator developments culminates in the need for a reusable infrastructure which can be used as a basis for each simulator.

This paper describes a major new simulation infrastructure package, the Software Infrastructure for Modelling Satellites (SIMSAT). It outlines the object oriented design methodology used, describes the resulting design, and discusses the advantages and disadvantages experienced in applying the methodology.

Key Words: Simulation, Object-Oriented, Satellite, Ada

1. INTRODUCTION

SIMSAT is being implemented by the European Space Agency (ESA) at its European Space Operations Centre (ESOC) in Darmstadt, Germany, utilising three software companies. The software development team for the project comprises ten staff, split between the three companies. Cray System's team of 5 software and simulation engineers is responsible for the real-time kernel of the system.

2. ROLE OF SATELLITE SIMULATORS

Simulation plays a vital role in the successful operation of satellites. Simulators are used:

- o to test the overall satellite control system, providing a realistic set of responses to the developers of such systems
- o to train spacecraft controllers, allowing routine and emergency situations to be realistically portrayed
- o to validate operational procedures, ensuring that inappropriate commands are not sent to the satellite, in both nominal and contingency situations

At ESOC the satellite simulators generally need to provide ground control centres with real-time telemetry data, via appropriate telemetry links. This must describe the dynamics, operation and status of the satellite and its subsystems, in the same manner as the real telemetry data which the control centre receives from the satellite. The simulator also needs to model the satellites response to telecommands.

In normal operation, the operations control centre (OCC) communicates with the satellite via a number of ground stations.

Alternatively a link is made to the software simulator. The simulator provides realistic modelling of the ground stations, the satellite orbit and environment, the operation of the satellite subsystems and payloads and their response to telecommands. The satellite also generates housekeeping telemetry. This is illustrated in Figure 1.

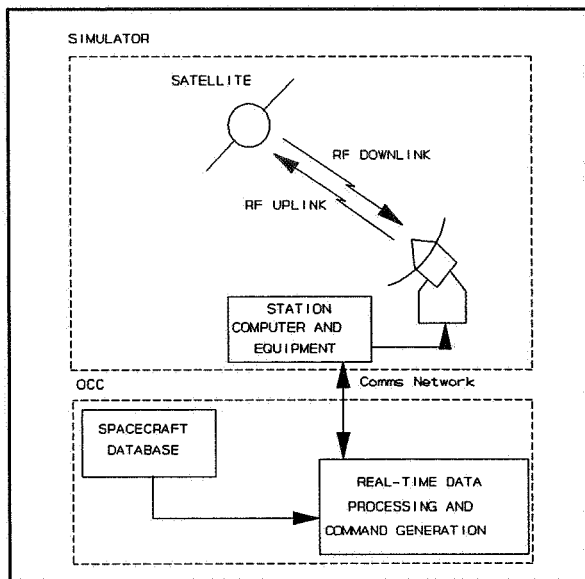


Figure 1 Typical simulator use

The simulator provides various facilities: injection of equipment failures and other events into the simulation (which are manifested to the operator by appropriate changes in telemetry), saving of data sets so that simulations can be started from particular points in the mission and monitoring of information on the health and status of the simulation.

Some simulators also include facilities to allow the on-board flight software to be run as part of the simulation.

As can be seen by the simulator roles described above, a high-fidelity simulation of the satellite and its subsystems is required, in order to give the operators confidence in the telemetry they receive. However the system information required to build a simulator of such fidelity is not available until relatively late in the development life cycle; yet the simulator must be available at least a year before launch, so that it can provide the required support for the necessary planning, testing and training. This means very short timescales for simulator development, hence the need for very high productivity and a sufficiently flexible implementation to accommodate late changes in specifications and requirements.

Since there is significant commonality between different satellite simulators, productivity gains are achieved through providing a common reusable simulator infrastructure which can be used as the

basis for each simulator together with a number of general purpose data-driven satellite system models. These include a model of the satellite orbit and the environment within which it operates. Thus the particular satellite simulator developer can make use of generic equipment models tailored by data derived from the latest system documentation, and can reuse (in a realistic manner) previously developed components.

3. OVERVIEW OF SIMSAT

The concept for SIMSAT is to provide those parts of simulators which are application independent as a general purpose system.

Figure 2 shows a typical SIMSAT-based simulator configuration.

3.1 Software Facilities

The operational "kernel" provides general purpose facilities for simulator control, scheduling, error logging and data management. Data Management includes data recording, providing data for display, real-time data access and saving of data sets.

The ground segment simulation includes common equipment models and interfaces to the OCC communications link.

There is a window-based graphics man-machine interface (MMI) used to display simulator information and to control the simulator, and also a reduced-functionality video terminal (VT) display system which provides rudimentary display and control facilities.

The major interfaces between SIMSAT and the satellite-specific code and the position and environment models are the Model Shell and SIMSAT Shell.

All services available to developers are concentrated in the SIMSAT Shell. These include services to schedule models, send telemetry, log events, and obtain key simulator information. These services are implemented using services from other SIMSAT objects, but this is transparent to the developer. This approach insulates the developer from any changes to SIMSAT, and reduces the complexity of the interface with which he has to work.

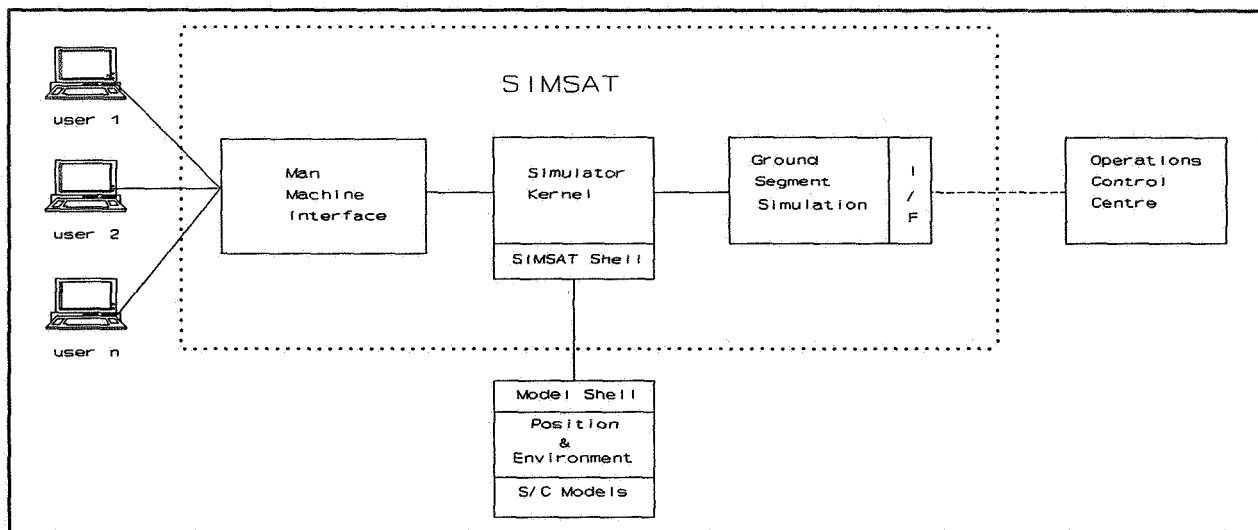


Figure 2 SIMSAT-based Simulator Configuration

The Model Shell is used by SIMSAT to notify the models of the following: change of simulator mode; command received; telecommand event; telemetry event; and model to be executed. The developer then provides an interface layer between this and the satellite-specific models.

SIMSAT will be interfaced to various standard models. These include telemetry encoding and telecommand decoding, power models (solar arrays and batteries) and an orbit and environment model, which includes simulation of the motion of the major celestial bodies (sun, earth, moon) and the perturbations to the satellite orbit caused by these bodies.

In addition there will be tools to assist with modelling of thermal control systems, attitude control systems and power distribution networks.

Simulator developers will then provide satellite-specific models to complete the simulation of their satellite.

3.2 Hardware Configuration

SIMSAT is designed to run on a DEC VAX Cluster. VAXStations host the MMI and/or simulator; a dedicated project computer is used to run the simulator when project resource requirements exceed those provided by a workstation. A file server provides reference versions of SIMSAT and the simulators.

Simulators can be configured in a variety of ways using SIMSAT, varying from running all components on a single workstation, to running the MMI, Ground segment and Kernel/Models on separate nodes of the cluster. Plans are also underway to enable models to be distributed across the cluster.

4. SIMSAT DESIGN RATIONALE

The need for reusability, together with the advantages of data-hiding and encapsulation led to the decision to implement SIMSAT using object-oriented design techniques. The anticipated benefits were as follows:

- o security could be enhanced since object access is only through its operations; access by simulator developers to the internals of the infrastructure parts of the system would not be possible
- o reuse could be enhanced, since the object-oriented approach leads to software modules which are easier to verify.
- o maintainability could be improved, since the well-defined interface specification and well-characterised behaviour of the objects would enable more rapid confirmation that any changes made had not affected the operation of the object or the system.

- o implementation of certain objects could be deferred, by hiding the implementation details within the object. The remaining objects could be developed on the basis of the declared interface specification, thus allowing the various parts to be developed independently of each other.

5. SIMSAT DESIGN METHODOLOGY

First the system level data which would form part of the ultimate simulator system was determined from the System Requirements Document. Objects were then identified whose purpose was to manage this data. Access to data was deemed possible only through the operations, termed services, of the object. The details of these services, required both to access the data and to initiate the behaviour of the objects, were then defined. Thus the implementation and structure, of both the data and the behaviour of the object, were not important (or visible) to the clients of the objects.

This approach resulted in 43 system level objects. In order to simplify the management and interaction of these objects, they were subsequently grouped into nine major components, and the interfaces between these components documented in a system level design document and interface specification.

Once the major components had been defined, more specific architectural design was then performed on each, including mapping each object on to the run-time processes. The objects could then be developed relatively independently, the interface specification being used to ensure continued compatibility.

6. OVERALL SIMSAT DESIGN

The resulting SIMSAT design comprises the following major components:

6.1 MMI Display

This is the full functionality display system. It includes objects which provide tabular, graphical and synoptic views of simulator information, which control the configuration of the simulator and which manipulate the log and the event schedule. It also provides commanding facilities.

6.2 VT Display

The reduced functionality display system, with no graphics capability, but able to display data in tabular form, and providing simulator control facilities.

6.3 Display Library Manager

Objects shared between the MMI and VT Displays.

6.4 Ground

Objects which support ground network functions. This includes the **Ground Controller**, the **OCC Simulator** and **Ground Equipment Models**.

6.5 Real Time Nucleus

Objects which provide the real-time core of the simulator: these comprise the **Mode Manager**, which controls the state of the simulator; the **Scheduler**, which together with the **Time Keeper** allows models to be scheduled in real-time; the **Telecommand/Telemetry Streams**, which provide the interface between the ground models and the satellite models; and the **SIMSAT Shell**.

6.6 Command

Objects used to control the simulator: the **Simulator Configuration Manager**, which starts and stops the simulator; the **User Status Manager**, which monitors the status of users connected to the simulator; the **Command Handler**, which allows the user to control the simulator; and the **Command Procedure Interpreter**, which provides facilities for executing groups of commands.

6.7 Data Management

Objects responsible for general management of data: the **Logger**, which records key events in a central history file; the **Public Data Manager**, which provides real-time data access and modification facilities, and can also save sufficient simulator data to allow the simulation to be restarted from the point at which the data was saved; and the **Data Recorder**, which provides recording and playback facilities.

6.8 Models

This component includes the **Model Shell** object.

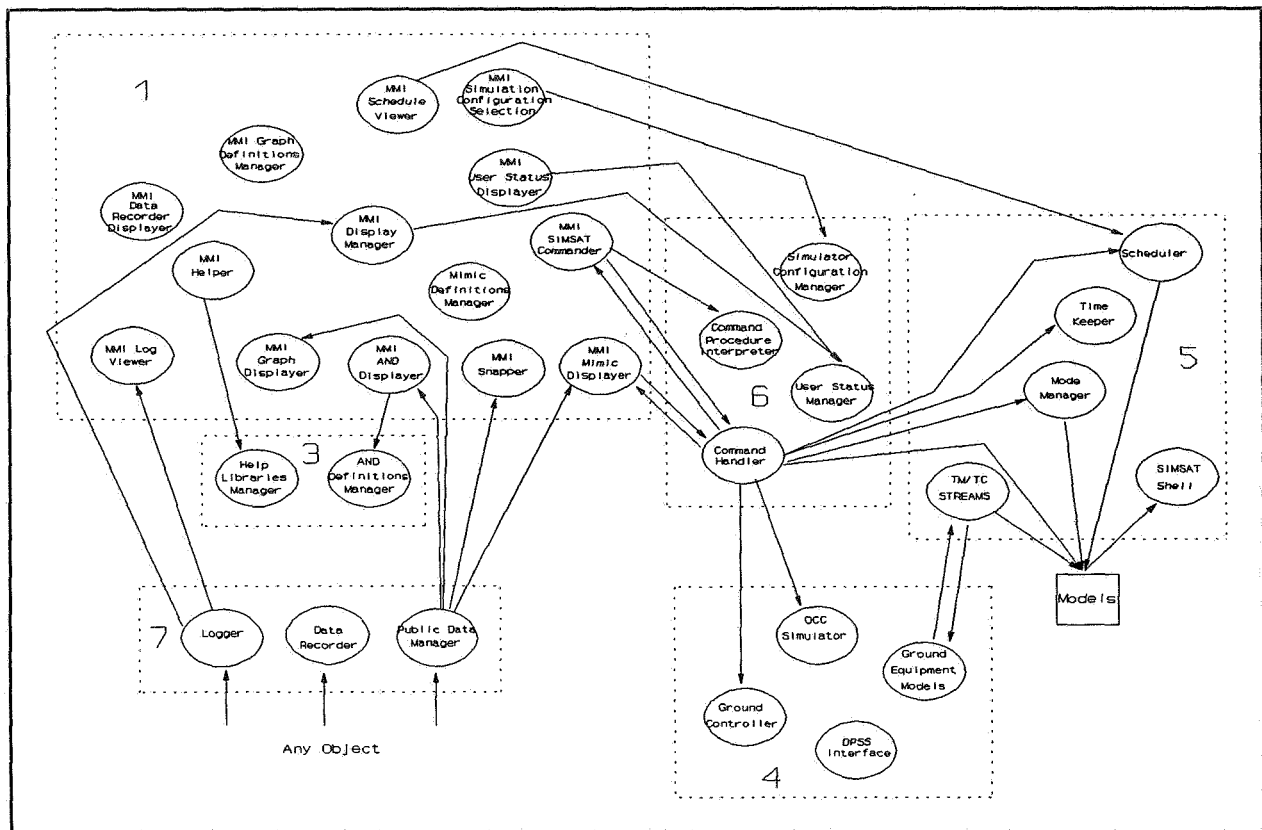


Figure 3 SIMSAT Objects and Service Usage
(Courtesy of ESOC)

6.9 Tools

The component comprises tools which are not internal to any other object. This is principally the **Spacecraft Database Reader**.

6.10 Object Interaction

Figure 3 shows the interactions between the objects, with the major components superimposed. Note that this diagram only shows service usage across major component boundaries. Service usage within major components is regarded as a lower-level effect.

The example shown is for the MMI; the VT case would be similar, since the VT and MMI are simply two examples of the class **User Interface**. Off-line tools are not shown.

Although object services may be used by any other object, there is in fact a relatively low degree of coupling between the major components of the system.

7. CONCLUSIONS

7.1 Benefits of Design Methodology

This section summaries the actual benefits experienced through the use of the object-oriented methodology.

7.1.1 Design Clarity

Specifying objects in terms of their data and operations quickly and clearly revealed any inadequacies in the system design, such as data which was not provided to objects which had need of it, or missing operations which were required. This allowed designers to resolve sooner the issues arising from such problems, when later they might be much more costly or difficult to solve.

7.1.2 Enhanced Security

Since clients did not have access to and were not able to make assumptions about the internal

implementation of an object, the integrity of the object. was maintained. This also insulated system elements from changes within other elements.

7.1.3 Phased Implementation

Since the objects were relatively independent and the work of each team did not need to be tightly coupled to the work of the other teams, a phased implementation of major components of the system was possible.

7.2 Drawbacks of Design Methodology

This section outlines the main drawbacks experienced.

7.2.1 Lack of Dynamic Visualisation

It was difficult to visualise the dynamic behaviour of the system, both in terms of the general interaction of objects and the specific data traffic loads between them.

7.2.2 Ada Constraints

The overheads associated with full data-hiding were not always acceptable. However, this was only of critical importance within the Real-Time Nucleus and certain parts of the Public Data Manager.

The concept of inheritance was not used as part of the implementation. This was because Ada only supports inheritance (and then to only one generation) through the use of types and generics.

There were extra development overheads incurred when extending the visibility of data internal to an object, since specific object services had to be provided.

The above points reflect the fact that Ada could be described as an object-based language, not an object-orientated language.

7.2.3 Documentation

It was necessary to define new documentation standards which suited the object-oriented approach and also met the ESA Software Engineering Standards.

7.3 Summary

SIMSAT is currently still being developed, and therefore it is not possible to report finally on the outcome of using object-oriented design techniques. However, to date, the anticipated benefits of using an object-oriented approach appear to have been confirmed.

An initial version of SIMSAT has been created, and a nominal test simulator defined. This is currently being used to test and verify the system.

The first delivery of SIMSAT is due in the first quarter of 1993, when it will be used in the development of a simulator of the CLUSTER group of satellites, which are being launched in 1995.

8. ACKNOWLEDGEMENTS

The author would like to thank J-J. Gujer and J. Miro for their assistance and general guidance.

Adam Williams is the Project Manager of the SIMSAT Kernel development team, and Cray Systems' Simulations Group Manager at ESOC.

SESSION J

ENGINEERING DATA ANALYSIS OF SPACE VEHICLE AND GROUND SYSTEMS

CO-CHAIRPERSONS

C. Kloss, Jr., JPL, California Institute of Technology, USA
W. R. Ochs, NASA Goddard Space Flight Center, USA

Summary	719
C. Kloss, Jr.	
J.1 Using SFOC to Fly the Magellan Venus Mapping Mission	721
A. W. Bucher, R. E. Leonard, Jr., and O. G. Short, Martin Marietta Astronautics Group, Denver, Colorado, USA	
J.2 Long Term Trending of Engineering Data for the Hubble Space Telescope	727
R. M. Cox, Computer Sciences Corporation, Seabrook, Maryland, USA	
J.3 High Performance, Low Cost, Self-Contained, Multipurpose PC Based Ground Systems	733
M. Forman, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA; W. Nickum and G. Troendly, General Electric Government Services, Inc., Greenbelt, Maryland, USA	
J.4 Expert Diagnostics System as a Part of Analysis Software for Power Mission Operations	739
J. A. Harris and K. A. Bahrami, JPL, California Institute of Technology, Pasadena, California, USA	
J.5 GenSAA: A Tool for Advancing Satellite Monitoring with Graphical Expert Systems	745
P. M. Hughes, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA; E. C. Luczak, Computer Sciences Corporation, Beltsville, Maryland, USA	
J.6 The Evolving Trend in Spacecraft Health Analysis	751
R. L. Kirkpatrick, JPL, California Institute of Technology, Pasadena, California, USA	
J.7 Orbital Signature Analyzer (OSA): A Spacecraft Health/Safety Monitoring and Analysis Tool	757
S. Weaver, C. DeGeorges, and J. Bush, Computer Sciences Corporation, Laurel, Maryland, USA; R. Shendock, TRW, Incorporated, Redondo Beach, California, USA; D. Mandl, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA	

SUMMARY — SESSION J

C. KLOSS, JR.
Jet Propulsion Laboratory

As the complexity of modern computer-based spacecraft increases, it is increasingly important that improved engineering data analysis tools be adopted. This session focused on improved data analysis tools, with the speakers showing how they successfully applied improved these tools to ongoing projects.

The speakers, topics, and projects were as follows:

- A. W. Bucher: Using SFOC (Magellan)
- R. M. Cox: Long-term trending (Hubble Space Telescope)
- M. L. Forman: PC-based ground systems (Nimbus/TOMS)
- J. A. Harris: Expert diagnostics system (Galileo)
- P. M. Hughes: Advanced satellite monitoring (COBE)
- R. L. Kirkpatrick: Spacecraft health analysis trends (Mars Observer)
- S. Weaver: Orbital signature analyzer (GRO)

Some key points that came out of the session:

- Designers of the tools need to work closely with the users.
- The analysis tools need to be cost effective and adaptable. For example, the SFOC at JPL is up to version 18.
- Simple portable workstations capable of interacting with the project from anywhere in the world are the wave of the future. For example, the Cassini project will have 57 PI's across the United States and Europe receiving data and sending commands.
- For complex spacecraft systems, rule-based, inferred graphical diagnostics with sub-level analysis capability will be needed.

The papers presented in the engineering data analysis of space vehicles and ground systems session have shown that there are multiple solutions to achieving success in the simplification of complex data analysis tools.

N94-23941

USING SFOC TO FLY THE MAGELLAN VENUS MAPPING MISSION

Allen W. Bucher, Robert E. Leonard Jr., Owen G. Short

Martin Marietta Astronautics Group, Denver, Colorado 80201, U.S.A.

ABSTRACT

Traditionally, Spacecraft Flight Operations at the Jet Propulsion Laboratory (JPL) have been performed by teams of Spacecraft experts utilizing ground software designed specifically for the current mission. The Jet Propulsion Laboratory set out to reduce the cost of spacecraft mission operations by designing ground data processing software that could be used by multiple spacecraft missions, either sequentially or concurrently. The Space Flight Operations Center (SFOC) System was developed to provide the ground data system capabilities needed to monitor several spacecraft simultaneously and provide enough flexibility to meet the specific needs of individual projects.

The Magellan Spacecraft Team utilizes the SFOC hardware and software designed for engineering telemetry analysis, both real-time and non-real-time. The flexibility of the SFOC System has allowed the Spacecraft Team to integrate their own tools with SFOC tools to perform the tasks required to operate a spacecraft mission. This paper describes how the Magellan Spacecraft Team is utilizing the SFOC System in conjunction with their own software tools to perform the required tasks of spacecraft event monitoring as well as engineering data analysis and trending.

Keywords: operations, ground data systems, telemetry processing, automation, Magellan.

1. SFOC INTRODUCTION

The Space Flight Operations Center (SFOC) is a collection of hardware and software subsystems working together in a multi-step process to provide mission control, telemetry processing, and storage services for a portion of the Jet Propulsion Laboratory's (JPL's) space flight missions (Ref. 1). SFOC receives spacecraft telemetry collected by the Deep Space Network (DSN) and performs frame synchronization and channelization of the telemetry.

Once the telemetry is channelized, it is distributed to the user for real-time display and interpretation. This channelized telemetry is also archived to allow retrieval and analysis at a later date. All the basic tools necessary for a user to display real-time telemetry and retrieve archived data are provided by SFOC.

2. MAGELLAN

The Magellan Spacecraft is a National Aeronautics and Space Administration's (NASA) planetary mission designed to obtain high-resolution Synthetic Aperture Radar (SAR) images of Venus. Magellan was launched aboard the Space Shuttle Atlantis on May 4, 1989 and using the IUS, was placed in a type IV transfer orbit arriving at Venus on August 10, 1990. Magellan began SAR imaging of the Venusian surface on September 15, 1990 and to date has produced high-resolution imagery of over 99% of the planet's surface.

3. MAGELLAN and SFOC

SFOC was designed to be the first true multi-mission ground data system to operate at JPL. It supplies core ground data system capabilities with enough flexibility to adapt to specific mission needs in a timely and cost effective manner. The first user of SFOC was the Magellan mission. The Magellan flight team became involved early in SFOC development and helped to provide requirements, testing and feedback during the initial releases of each SFOC version. This led to the Magellan ground system configuration depicted in Figure 1. The first version of SFOC used by the Magellan spacecraft team was Version 6. This version was used to support some of the pre-launch spacecraft testing and spacecraft team training exercises. The next version, Version 7, was used to support launch and the majority of cruise operations. Magellan is currently using Version 13, which came on-line just

prior to Venus orbit insertion. The latest versions of SFOC are being developed under the new Advanced Multi-Mission Operations System (AMMOS) program.

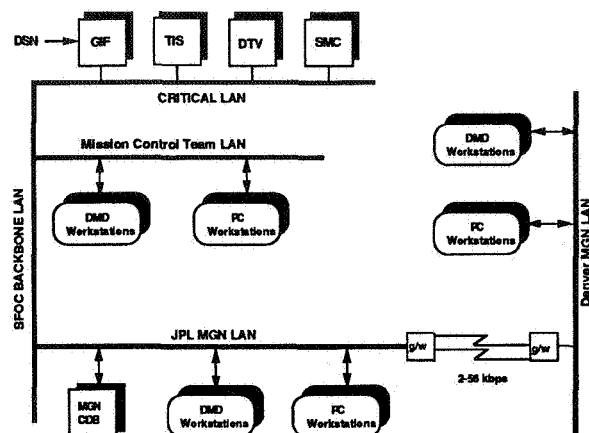


Figure 1 Magellan Ground Data System

4. SFOC REAL-TIME CAPABILITIES

SFOC provides the following real-time telemetry monitoring capabilities:

- Data Monitor and Display (DMD),
- Derived telemetry channels,
- Alarm limit maintenance and usage,
- Latest available data dump requests,
- Real-time memory readout verification.

4.1. Data Monitor and Display (DMD)

The Magellan spacecraft team uses the SFOC Data Monitor and Display (DMD) capability to monitor real-time telemetry via the spacecraft team workstations. Each subsystem has customized its DMD environment based on the characteristics of the individual subsystem and personnel. The DMD function allows engineers to define their own display screens or templates using the Template Definition Language (TDL). Although simplistic in capability, TDL provides the user with the tools required to create imaginative displays. The display types used by the Magellan spacecraft team include Fixed, List, Matrix, Message, Alarm and Plot pages.

Fixed Pages: These display types are designed by the subsystem engineers to logically group data in pictorial form which best suits their monitoring needs. The "fixed" indicates that the formats and data assignments cannot be modified in real-time.

This page type is preferred by the Magellan spacecraft team for real-time data monitoring. An example of a Magellan power subsystem fixed display is depicted in Figure 2. The Magellan spacecraft team has designed and coded in excess of 100 fixed pages for flight use.

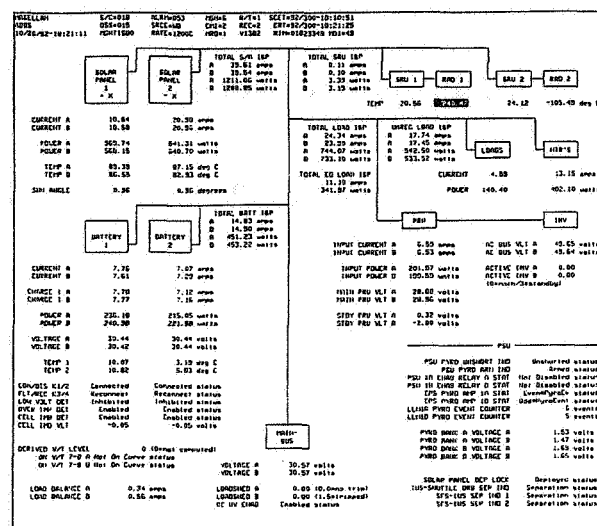


Figure 2 DMD Fixed Display

List/Matrix Pages: The data contents of list and matrix pages are designed to be modified in real-time simply by modifying a list of data channels. The data attributes to be displayed such as titles, values, time, units, etc are designed to be displayed on a single line for a list page and in a grid for matrix pages. This display type is useful when a subsystem wants to change data content on the fly.

Message Pages: The message page is designed to emulate a line printer on a video display. As new channel messages appear, they scroll down the screen. The format of the message page is user definable allowing the individual to display data attributes of interest. Message page output can also be routed to printers or data files.

Alarm Pages: This page type is designed to display channel information of telemetry channels which exceed a predefined threshold. The alarm page is structured like a list page. When a channel goes into alarm it is added to the alarm page and it maintains its position as data is updated.

Plots: For subsystems that monitor analog data, the capability to plot telemetry real-time has been extremely valuable. Plot types include Channel vs. Time, Channel vs. Channel and Minimum/Maximum

plots. The DMD is capable of displaying 20 individual plot windows on a single workstation screen with up to four channel assignments per plot window. This allows up to 80 channels to be monitored graphically. The size of the plots, scales, labels and time ranges are all user definable.

Figure 3 shows a typical subsystem display configuration and contains fixed, list, alarm, and plot display types.

4.2. Derived Telemetry Channels

The DMD also provides the capability, via the Channel Conversion Language (CCL), to produce derived telemetry channels. Derived telemetry channels are measurements which are derived from information received in the main telemetry stream. The derived channels can be as simple as changing the units on a received value, to as complex as deriving the remaining propellant load on a spacecraft. In general, any data that is used for spacecraft analysis that can be generated from a snapshot of telemetry is a likely candidate for a derived channel. Just like the main telemetry channels, derived channels can be displayed, alarm checked, and plotted using the DMD. They can also be retrieved from the central database for non-real-time analysis or trending. The calculations of "Right Ascension and Declination" are simple but useful examples of derived channels developed by the Magellan spacecraft team. The algorithm for calculating the right ascension and declination utilizes the 4 quaternion elements telemetered from the spacecraft and is as follows:

$$c31 = 2 \times ((quat1 \times quat3) + (quat2 \times quat4))$$

$$c32 = 2 \times ((quat2 \times quat3) - (quat1 \times quat4))$$

$$c33 = quat4^2 + quat3^2 - quat2^2 - quat1^2$$

$$right\ ascension = ATAN(c32, c31)$$

$$declination = ASIN(c33)$$

The Magellan flight team has developed and implemented approximately 250 derived channels during mission operations.

4.3. Alarm Limit Maintenance and Usage

The capability exists within the DMD to check user-specified telemetry and inform the user when an unexpected value is encountered. This function is accomplished using DMD yellow and red alarms. The yellow alarms are set and monitored by subsystem engineers locally at each individual workstation and are generally used as caution alarms. It requires only subsystem lead approval to change yellow

alarms. The red alarms are monitored on a system wide basis and usually indicate that an undesirable/take-action event has occurred. They are monitored round-the-clock by Magellan mission control team specialists. Red alarm values have been typically selected utilizing the following guidelines:

- Adherence to launch/deploy go/no-go criteria,
- Adherence to specified acceptance test limits,
- Adherence to S/C fault protection thresholds,
- Predicted values for mission phases,
- Nominal S/C values observed pre-launch.

To insure the correct level of review, Team Chief level approval is required to change a red alarm value.

4.4. Latest Available Data Dump Requests

Another feature of DMD used by the spacecraft team is Latest Available Data (LAD) dump requests. Maintained in the workstation is a buffer which contains the latest available data of all activated telemetry channels. The contents of the buffer may, at any time, be dumped to a printer or a file for further processing by spacecraft analysis tools. The Magellan spacecraft team used this capability to capture the state of the spacecraft following an anomaly which caused unexpected loss of downlink. One of the first actions performed after the anomaly occurred was a dump request of the LAD buffer. This LAD dump was immediately reviewed by the subsystem engineers for clues to the cause of the anomaly.

In addition to LAD dumps, the capability to perform channel parameter table dumps and coefficient dumps is also available. These capabilities can be used to determine channel information such as current alarm limits or polynomial coefficients.

4.5. Real-time memory readout Verification

There have been several instances during the mission where the spacecraft team has had to make flight software modifications or parameter changes that require real-time Memory ReadOut (MRO) verification. The majority of these software modifications are performed by a sequence of command groups. In general, memory verification of one command group is required before the next command group may be sent. This verification was usually required in a time-critical manner. To allow timely MRO verification, SFOC developed a tool to perform real-time MRO verification. This tool, named BCTOMRO, strips MRO data from the

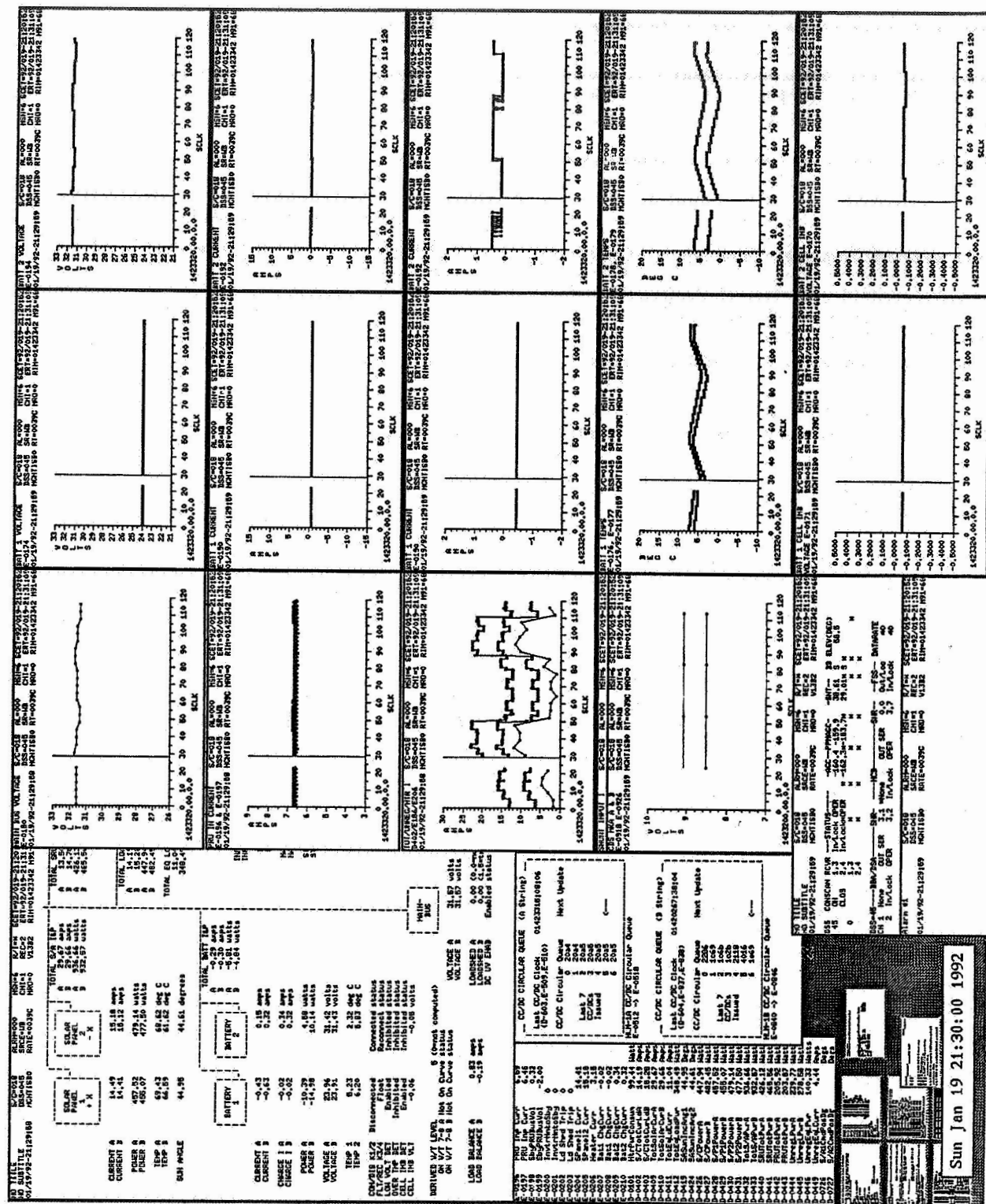


Figure 3 Typical Magellan Subsystem Display Configuration (Power Subsystem Battery Recondition Display)

SFOC telemetry stream and displays the data in real-time. BCTOMRO has allowed the spacecraft team to make the real-time MRO verifications of changes and continue commanding during time critical command windows.

5. NON-REAL-TIME DATA ANALYSIS

SFOC has provided several tools to aid in non-real-time analysis of spacecraft data. The spacecraft team uses these tools in combination with tools they developed to perform the following non-real-time functions:

- Telemetry recall from the central database,
- Automated analysis and trending,
- Telemetry state verification,
- MRO data recall from the central database,
- Analysis of spooler files.

5.1. Telemetry Recall from the Central Database

The most important SFOC tool developed for data analysis on the Magellan program has been the capability to query data from the Central DataBase (CDB). The Magellan CDB consists of 45 days of on-line storage of channelized spacecraft engineering telemetry, DSN monitor data and associated ancillary data channels. The query capability has been set up so that any engineer from a SFOC-provided workstation can extract data concerning the spacecraft or ground activities. The SFOC-provided tools used by the spacecraft team for querying the CDB include DMDETf, DMDQMOD, and DMDQUERY. These tools allow a user to produce "canned" queries that are used repetitively or "ad-hoc" queries that are built and submitted in minutes. The user selects the desired telemetry channels and the analysis interval. The data is returned in an ASCII Data Return File (DRF). This DRF can then be input to any one of numerous analysis programs or may be used as a stand-alone tool for data analysis.

5.2. Automated Analysis and Trending

Although the CDB query capability itself has been extremely useful, it became apparent that the Magellan spacecraft team analysts were performing a variety of repetitive queries and analysis during their daily and weekly routines. To make better use of an analyst's time these repetitive tasks have been automated using Unix-based scripts. This automation allows the analyst to perform additional mission analysis and anomaly resolution. The majority of the automation tools consists of scripts written in

AWK, PERL, CSH, SH, etc, ... and some "C" code. The automated tasks were derived from manual tasks allocated to each subsystem in flight team procedures. The tasks were performed manually until they were well understood and then the engineers constructed a tool to carry out the task. It is important to note that the subsystem engineers performed their own coding since the spacecraft team did not have a dedicated software staff. One example of an automated script is the engineering telemetry long-term trending script. The first step in this script is to query the desired analog telemetry (temperatures, voltages, currents, etc.) over a specified interval, usually 24 hours. The script uses the resultant DRF to calculate the minimum, maximum, average and standard deviation for each telemetry channel over in the entire length of the file. These statistics are then appended, along with a time stamp, to a set of long-term channel trend files. There is a channel trend file for every telemetry channel of interest. The result is a set of long-term trend files that contain trending statistics for each 24 hour period from the beginning of the mission for each channel of interest. These trend files are formatted like DRFs which allows them to be plotted and analyzed using the same tools used to plot and analyze DRFs. Therefore, the trend files are used for analysis and are periodically plotted against long-term predictions using VMPLLOT. An example would be to plot the actual long-term solar panel maximum current trend versus predicted solar panel current to quantify panel degradation over the life of the mission. The process outlined above has been fully automated with use of "perpetual" scripts. A "perpetual" script performs its task and then re-submits itself to run again at the next specified interval, without any operator intervention. For instance, once a day an automated script queries the desired channels to be trended and re-submits itself to run the next day. One other advantage of the automation scripts is their ability to run during early morning hours when system loading is low.

5.3. Telemetry State Verification

Originally Magellan did not have the capability to autonomously track spacecraft states. All critical component state tracking was accomplished by engineers from each subsystem manually recording their individual subsystem states in a spacecraft state table. However, soon after launch, the spacecraft team developed the ability to autonomously extract all spacecraft states that are telemetered. The spacecraft team created a Unix script entitled "Moose Query" which queries all Magellan

telemetry and produces a summary report of all spacecraft states. Although this report is still compared manually to a table of expected spacecraft states, it greatly simplifies collection and recording of the approximately 1000 spacecraft states.

In addition to the state reports, moose query also produces plots of all analog data and tabular printouts of all status information over a user selectable time range. Since all spacecraft telemetry is either plotted or printed, this tool proved valuable during anomaly investigations. Following critical anomalies, a moose query would be submitted and the engineers would receive a copy of the output within hours for review. It should be noted that many anomalies were resolved by data from telemetry channels showing anomalous indications but were not immediately thought to be related to the anomaly.

5.4. MRO Data Recall from the Central Database

In addition to storing engineering telemetry, the CDB also has the capability to store MRO data for retrieval at a later date. The MRO query utility allows the engineer to retrieve memory readouts received from the spacecraft for any of the on-board computer memories. The user supplies the desired memory device, the start and end addresses, and the begin and end times of MRO collection. The MRO utility queries the CDB and returns an ASCII file identifying address and the memory contents at that address. This return file is utilized by the spacecraft team analysis programs, MEMANAL, VMPTRK, CHKSUM, and VMFLOAD to verify the contents of on-board memory and to verify flight software parameter values. Once the MRO is verified to be correct, the file is catalogued as the current updated ground memory mask. This mask is used as a baseline if an on-board computer memory ever needs to be reloaded. This process was used to reload AACS "Memory B" after an address bit failed in one memory bank shortly after Venus orbit insertion.

5.5. Analysis of Spooler Files

SFOC can also produce spooler files of telemetry data. These spooler files represent a collection or a stream of telemetry for a selected time period. The files are expanded to include all telemetry and ancillary information such as header data, monitor data, quality flags, etc. The processing of these spooler files is most easily done using a SFOC-provided utility called BROWSER. BROWSER allows an engineer to review spooler files and analyze the content for correctness. BROWSER has the capability

to build templates that a) display data in a certain orientation, or b) filter through a spooler file displaying only the data that match predefined conditions. The capability to produce and analyze spooler files has proven useful for the Magellan flight team when troubleshooting problems with the telemetry stream. For example, the use of spooler files and BROWSER allowed spacecraft analysts to isolate and provide solutions to a radar telemetry spiral wrap anomaly. A "raw" spooler file and a "channelized" spooler of the same data were compared using BROWSER. From this analysis it was concluded that the ground decommutation process was working correctly and the problem was isolated to a timing problem within the radar, which has since been corrected. BROWSER was also used extensively to isolate the on-board tape recorder problems.

6. SUMMARY

SFOC has provided some extremely powerful analysis tools. The DMD provides enough flexibility to meet the dynamic real-time analysis needs of each individual subsystem and, although considered "Expert Friendly", it does not constrain the user as do many sophisticated graphical user interfaces. The capabilities and flexibility of the CDB query process have significantly contributed to the success of the Magellan mission. The versatility and flexibility of the SFOC tools and Unix have allowed the spacecraft team to build software and scripts "on top of" SFOC tools to provide automation and remove manual burdens from the analysts. Although the SFOC tools experienced some growing pains, the SFOC development staff has always responded well to problems and suggestions. Being involved in the development and use of SFOC has been a rewarding experience.

7. REFERENCES

1. Miller, D., Palkovic, L., Edwards, C., "Space Flight Operations Center User's Overview", July 1991, pages 1-6.

8. ACKNOWLEDGEMENT

The work described in this paper was accomplished by Martin Marietta under contract to the Jet Propulsion Laboratory, California Institute of Technology and sponsored by the National Aeronautics and Space Administration.

N94-23942

LONG TERM TRENDING OF ENGINEERING DATA FOR THE HUBBLE SPACE TELESCOPE

Ross M. Cox

Computer Sciences Corporation
Seabrook, Maryland, USA

A major goal in spacecraft engineering analysis is the detection of component failures before the fact. Trending is the process of monitoring subsystem states to discern unusual behaviors. This involves reducing vast amounts of data about a component or subsystem into a form that helps humans discern underlying patterns and correlations. A long term trending system has been developed for the Hubble Space Telescope. Besides processing the data for 988 distinct telemetry measurements each day, it produces plots of 477 important parameters for the entire 24 hours. Daily updates to the trend files also produce 339 thirty day trend plots each month. The total system combines command procedures to control the execution of the C-based data processing program, user-written FORTRAN routines, and commercial off-the-shelf plotting software. This paper includes a discussion the performance of the trending system and of its limitations.

Keywords: trend, processing, software, analysis, plotting, spacecraft

1. INTRODUCTION

Many spacecraft have been designed with expected life times of 3 to 5 years. Even for such short term missions, both the quantity and quality of personnel available for spacecraft engineering operations decline as the mission ages. Budget constraints, individual career advancements, and contractor changes all influence this decline. As mission lifetimes now extend to decades, the need to transmit knowledge concerning the behavior of hundreds of spacecraft components becomes critical. Trending is a useful tool in this transfer. It allows future subsystem engineers (SEs) to have access to the data their predecessors deemed pertinent to proper operation of the spacecraft. The establishment of valid trends also helps an SE become familiar with the way the spacecraft actually operates on-orbit, which is not always as originally designed.

While most SEs acknowledge the need for trending, each has a different idea of the best way to analyze a spacecraft. One way to differentiate between different types of spacecraft analyses is on the basis of where the processing takes place. In this paper, *Online Analysis* is assumed to occur on the same system that takes in realtime data. The realtime support schedule is presumed to be the sole driver for performance of this analysis. *Offline Analysis* is defined as that which goes on despite the availability of the spacecraft.

The duration of the analysis provides another distinction between different types. *Realtime Analysis*, performed during critical activity periods has a duration measured in seconds or minutes. The term realtime is derived from the operations of low earth orbiters, when available commanding time as limited. The concept can be extended to include geosynchronous operations as applied to critical operations such as orbital maneuvers or high precision pointing operations. The basic purpose of realtime analysis is to verify that all expected events occurred. Nominally for prevention of unexpected or hazardous conditions and to quickly detect anomalies, a major drawback to realtime analysis is the many operations proceeding concurrently. The detection of more subtle problems that arise is difficult.

Near-Realtime Analysis, as the name implies, occurs shortly after a realtime contact. Its time scale is from minutes to hours, and its purpose is usually to investigate further some unexpected state or condition recently observed in one or more realtime contacts. It allows for greater correlation between different data points, since it is carried out in a less hurried environment.

Trending is the aspect of offline analysis used to reduce the quantity of data saved to characterize component or subsystem performance. A big problem encountered in trending concerns the volume of data to be processed. Trending can therefore be broken down into short and long

terms. Short term implies data collected for hours or days at a time. Long term reflects data collected for weeks, months, and years. The short term trends can normally process every point received from the spacecraft, whereas data reduction strategies must be applied for long term trends to retain information within data storage limits.

This paper will describe issues encountered in the design and implementation of the Long Term Trending System (LOTTTS) used by the Hubble Space Telescope (HST). First an overview of the spacecraft, telemetry, and data flow defines the scope of the task. A brief description of the hardware and software environments is followed by discussion of the internal LOTTTS processing. Lastly, restrictions and performance considerations are presented.

2. OVERVIEW

A primary goal in creating LOTTTS was to provide the SEs with a useful tool that would help in producing required plots for Mission Operations Contractor (MOC) monthly reports. An underlying theme to the development was to design a system the user could easily modify to meet changing mission needs. Before characterizing the LOTTTS, the spacecraft and ground system components with which it interacts will be described.

2.1 Spacecraft Subsystems

The HST spacecraft has seven major subsystems: Data Management, Electrical Power, Instruments and Communications, Optical Telescope Assembly, Science Instrument Command and Data Handling (which has four elements, one for each instrument onboard), Pointing Control, and Safing. Each subsystem imposes unique requirements for trending in terms of the input data rate, frequency of desired output, and complexity of the processing involved. Most inputs to LOTTTS come from telemetry.

2.2 Telemetry

There are approximately 7000 different engineering measurements that may be placed into the telemetry stream. Each measurement has a User Friendly Mnemonic (UFM) of up to eight characters. For instance, DTR2MTRC identifies tape recorder 2 motor current. The UFM's remove the need to remember minor frame and word locations. The

ground system database handles the conversions from UFM to minor frame/word/bit. The telemetry is transmitted in at least one of seventeen data formats. The data rate is 4 or 32 kilobits per second (kbps). The telemetry matrix major frame is 1200 minor frames by 200 words for 32 kbps and 250 minor frames by 125 words in 4 kbps.

2.3 Data Flow from Spacecraft to LOTTTS

Data from the spacecraft, is processed through the Tracking and Data Relay Satellite System (TDRSS). It is received at the Space Telescope Operations Control Center (STOCC) at Goddard Space Flight Center (GSFC) located in Greenbelt, Maryland. For realtime operations, data then flows to the Payload Operations Real Time System (PORTS), which produces realtime history files. The PORTS Applications Software System (PASS) accepts the Near RealTime data (seconds to minutes time lag) from PORTS. After an engineering tape recorder dump, the recorded data, PORTS history files, and the NRT data are merged to form the Astrometry and Engineering Data Products (AEDP).

2.4 Engineering Subsets

PASS creates engineering subset files as part of the AEDP based on the contents of the Telemetry Subset Definition (TSD) file[1]. The subsets are written in changes-only format. This means that data is only written to the file if its value has changed since the previous minor frame. This serves to greatly reduce the data processing. The particular subset used by LOTTTS contains nearly 1000 telemetry points selected by the subsystem engineers. These data are converted to engineering units and time tagged.

3. HARDWARE ENVIRONMENT

The LOTTTS operates on computer systems residing in the Engineering Support Systems(ESS) room at GSFC. The hardware consists of a dedicated Digital Equipment Corporation (DEC) VAX 3100/30 workstation attached to an A/B switch. This switch allows connection into either the network of other ESS 3100 workstations or a cluster of DEC mini-computers that serve as applications processors (AP5/AP7). These reside in the Data Operations Control Center. With the switch in the "A" position, the workstation has access to the AP5/AP7 cluster. This is the normal daytime configuration. It allows SEs with accounts on AP7

to access the trend files or binary data files directly to perform special analyses. The data base and PASS products reside on the cluster. In the "B" position, the workstation can access an LPS20 laser printer to produce hardcopy.

The specifications on the hardware are: 1) the workstation operates at eight mips, has sixteen megabytes of RAM and 1 million blocks (1 block = 512 bytes) of storage, 2) the DEC cluster computers consist of one model 6510 and one model 8650. The 6510 is a 6 mips machine with 60 megabytes of RAM. The 8650 is a 12 mips machine with 64 megabytes of RAM. These two mini-computers share 31 gigabytes of storage. The LPS20 printer has a 1 megabyte onboard memory, processes 20 pages per minute, and has a turbo chip set to improve graphics processing time.

4. SOFTWARE ENVIRONMENT

Since trending is a routine event, it is well suited to a batch environment, requiring minimal operator intervention. LOTTS operates under the VMS 5.4 operating system. Digital Command Language (DCL) command procedures control the operator interface. The VAXC programming language is used for the programs that read the engineering subsets and update trend files. User programs are written in VAX FORTRAN. The plotting software uses a commercial-off-the-shelf product, IDL, developed by Research Systems, Inc.

5. LOTTS INTERNAL DATA FLOW

The data inputs to the LOTTS are the engineering subsets. AEDP creates them on the AP5/AP7 cluster. Trending begins with data technicians monitoring the creation of these files. PASS produces these files, as possible, throughout the day. When an entire day's files appear, the operator executes procedures that copy the engineering subsets to the trending workstation and remove any old data from the cluster. Next, three jobs start up in a batch queue. The first breaks the subsets into a separate 24 hour file for each telemetry point. The second job creates user-defined derived parameters using the output of the first job. The third job creates daily plots (see Figure 1).

5.1 Creating 24 hour UFM files

Each engineering subset file has the date-time of the first time tag it contains embedded in its name.

This file contains variable length records because of the changes-only nature of the data. The first record is a header from which the start time, stop time, and telemetry format of the file are extracted. Each format change causes the current file to close and a new one to open. Every UFM has an associated Measurement Tag Value (MTV), an index into the alphabetical list of UFM's. When writing a subset file, AEDP inserts a minor frame time, followed by an MTV/value pair for each point that has changed in that particular minor frame.

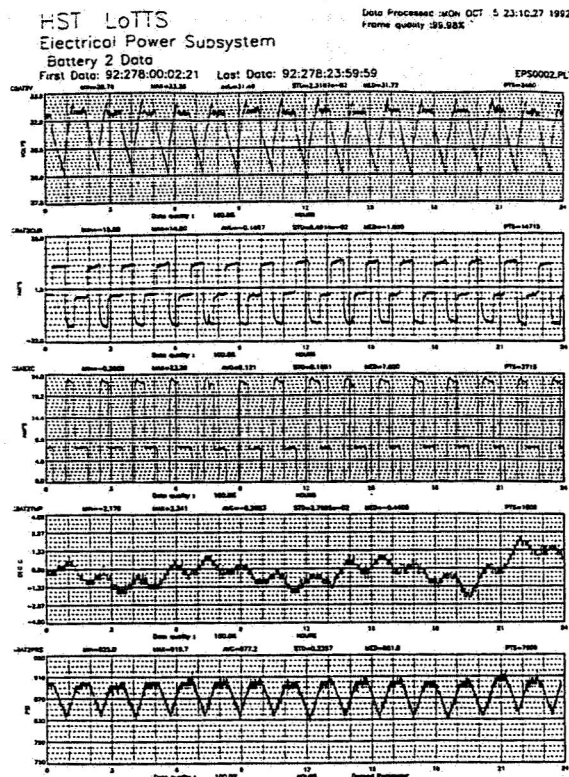


Figure 1 - Typical daily plot produced from LOTTS. It shows, from the top battery voltage, battery current, solar array current, battery temperature and battery pressure for battery two.

The subsets represent large files containing all data for all requested points. The 24 hour UFM process breaks these into many smaller files, each with all data for a single point, over an entire day. These products receive the name of the UFM's for easy identification. They are called bin files, as they are output in a flat binary format. Each bin file has 1536 byte records containing data and time tags. The first record is a header containing status data including the UFM, MTV, data start and stop times, date and time when the data was processed by

LOTTs, number of minor frames read, number of frames with bad quality, the total number of data points read, number of points that failed conversion, data type for this parameter, a descriptor to indicate any special processing of this point (See section 7), and the minimum, maximum and time-based average for the point. This information is later used to help in processing the file or as display data on a daily plot.

Other processing done by the 24 hour UFM process includes marking data dropouts. Each minor frame contains a quality flag. When this flag shows a problem with the data, the program places a large negative value in the file for the given time. This is necessary because the changes-only data produces ambiguity between constant data and no data. The large negative value allows the plotting software to induce a pen-up for dropouts. Note that problem data is not involved with statistics collection.

Error logs from each run show the time locations of data dropouts. These are used by the SE to determine if a data spike was the cause of a data hit or a component anomaly. A final feature of the 24 hour UFM process is the creation of a temporary statistics file. This is used by another program to update the trend files.

5.2 Derived Parameters

The second major processing function is to establish derived parameters. These increase the versatility of the LOTTs by allowing data to be presented in simpler forms. Trending applies to derived parameters just as to normal telemetry. Bin files are created and the temporary statistics file is updated.

Derived parameters are useful for several functions. One form of derived parameters is non-telemetered algebraic combinations of two or more telemetry values. A simple example is the creation of a power parameter using voltage and current telemetry. Derived parameters are also useful when the ground database contains an incorrect calibration curve. Databases should be under the strictest configuration control. In this case, updates will occur only after errors accumulate to the point of warranting a new version. With derived parameters, the SE creates the proper calibration routine until the new curves are implemented. Derived parameters are useful for coordinate transformations and unit conversions as well.

One very important use of derived parameters is made in the treatment of context dependent telemetry, where one telemetry point controls the validity of another, such as a component's telemetry taking on meaningless values when idle. Another example is the SE wanting to see the data only during a part of the orbit such as terminator crossings.

The greatest value of derived parameters in the LOTTs is that the SEs can modify, create or add algorithms at their discretion without extensive programmer intervention. Basic utilities are provided for reading and writing both ASCII and binary files and performing calibrations. There is single subprogram for each subsystem to handle its derived parameters. The SE simply creates a subroutine to perform the desired calculation and modifies the derived parameter subprogram to call it. This leaves LOTTs programmers to handle the more complex issues and prevents errors in one subsystem from propagating to the others.

5.3 Trending

After the subsets are processed and derived parameters are produced, the trending takes place. The subroutine takes the statistics file, generated by the 24 hour UFM and Derived Parameter processes, and updates the trend files. This occurs quickly since the trend files are indexed by year and day of year. The trend data is stored in 80 byte records and includes: minimum, maximum and average for the given day and year.

5.4 Plotting

Next, the daily plots are produced. A template file defines each plotted page. Concatenating many of these templates allows each SE to customize plot requests as the mission requires. Theoretically, there is no limit to the number of plots placed on a page, but the resolution becomes impaired after 8 in portrait mode (6 in landscape). The templates specify the number of plots per page, plot orientation for the page, UFM's of the points to be plotted, number of coarse and fine grids lines, upper and lower bounds on both axes, plotting symbols and connectivity, and limits. The user has the option of placing statistics for a grid just above the top line. These statistics include minimum, maximum, average, standard deviation, and number of points plotted. Pages can be labeled with major and minor titles as well. The user can select

automatic scaling based either on the minimum and maximum data values or n-sigma, where n defined by the user. The plotting software provides default values to all unsupplied parameters except the UFM and page numbers. The same templates are used for trend plots. An example is given in Figure 2.

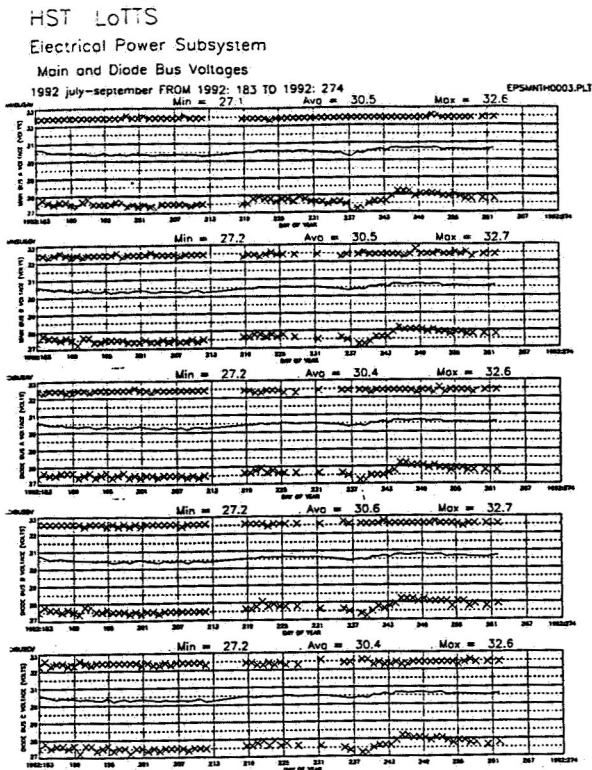


Figure 2 - A sample long term plot. Averages are connected dots, minimums and maximums are shown with unconnected Xs.

6. RESTRICTIONS

It was known from the beginning that certain restrictions would have to be placed on the users of the system. This is because, in general, users have the uncanny ability to consume all resources, in terms of CPU storage, etc., they are allowed. For LOTTS, this meant devising compromises to maximize engineering content while minimizing total processing.

6.1 Number of Telemetry Points Processed

If you ask an SE how much of the available data he or she would like to see, the response is invariably "All of it!". In practice, however, the SE looks at only a portion of the data on a routine basis. For instance, it is not uncommon for data on a backup

component to be meaningless when not in use. The backup component is still important, but the trended data is meaningless. The criteria established for allowing mnemonics into LOTTS was to impose a total limit of 100 per subsystem plus 100 for each instrument or 1000 total. The limit relies on the supposition that an engineer, in normal operations, would not use more than 100 points, (approx. 12 pages), for daily or monthly reports. The supposition comes from both past experiences with other spacecraft, and actual monthly report content.

The benefit of imposing a limit is that it forces each SE to seriously consider those points they really need and prevents them from simply asking for everything, much of which they would never use. Note that the limit applies only on the total number of points, not points per subsystem, so one group may be plotting or trending more than 100 points.

An important consideration when imposing limits, is that specific mnemonics must be easily added to or deleted from the total, such as when a primary fails or interest rises in a previously untrended point. LOTTS accommodates this through requesting changes to the PASS controlled telemetry subset definition file.

6.2 Culling Algorithm

Some data are telemetered at 40 hz. If one of these points were monitored every second over 24 hours it would produce over 3.4 million time-value pairs. Plotting such large files can take hours, with no real benefit, since the number of points is greater than the printer's resolution. The problem is how to throw out data without losing information content. The solution used in LOTTS, termed culling, is to use a time window of averaging. The user specifies time period over which the minimum, maximum, and average will be found for each offending UFM. The bin files will contain just these three points for each time period. Using this method, plots of over one million points reduce to near eight thousand points, with no visible information loss on the plots. Currently, less than ten points require this treatment, but the plot software is spared the need to plot over 9 million points a day.

6.3 Filtering Scheme

Another problem of using plots to analyze data is what to do with the spikes. They may be an

indication of an imminent failure, but most of the time the problem is in the data. Data spikes make automatically scaled plots take on unreasonable ranges and wreck statistical calculations.

LOTTTS solves this by removing the spikes from the plotted data, but providing a listing of their times and values. This is called *filtering*. There are two kind of filtering rate and absolute. In rate filtering, the SE defines a delta amount by which a point can change and still be plotted. Any changes greater than the delta will not be written to the plot file. Absolute filtering disallows any values over or under the user-defined range.

7. PERFORMANCE

7.1 Processing Time

The engineering subsets are available within 3 days (usually 1), of realtime. The delay is largely due to waiting for the tape recorder to be dumped. It then takes 2-4 hours to create the bin files and derived parameters. 1-1.5 hours are required to create plot files. And 2-2.5 hours are required before hard copies of the daily plots are in hand. This totals roughly the time of an 8 hour shift. Monthly plot creation takes 2-3 hours.

7.2 Short Term Storage

The normal telemetry data consists a mixture of 32 and 4 kbps data. The designed mixture was 20% 32 kbps and 80% 4 kbps. Due to mission changes since launch, the mixture is now closer to 80% 32 kbps and 20% 4 kbps. The size of engineering subsets for a given day ranges from 45,000 blocks to 120,000 blocks. A rule of thumb to process the data completely is that three times the space used by the subsets will be required. LOFTTS programs and permanently stored files require 200,000 blocks. This limits the amount of on-line storage of bin files to about three days[2].

7.3 Long Term Storage

A final performance issue on LOFTTS is due to size of the trend files themselves. Currently, these files take up around 67,000 blocks. Storing 1000 sets of statistics, per day, for 15 years will eventually consume over the 1,000,000 allocated to LOFTTS. As stated earlier, present operations allow the bin files to remain on the system for three days. Allowing for a doubling of LOFTTS and user defined

software, 400,000 more blocks will be used up. For these longer term trends it is not necessary to keep each days statistics. The solution is to compress the daily trend files into monthly statistical form. For LOFTTS, daily files for the last five years will be stored on-line and older data will be stored by month.

8. CONCLUSIONS

This paper has addressed various aspects of the LOFTTS as applied on HST. First, trending is performed as a routine activity that does not interfere with critical operations. Also, the trending and daily products are readily accessible to the subsystem engineers. The design incorporates the ability to change with mission needs. Several methods were presented to show that large volumes of data are successfully reduced, without loss of information content, for both analytical and storage reasons. Long range storage solutions have shown that the LOFTTS can last throughout the mission lifetime.

9. ACKNOWLEDGEMENTS

The author wishes to thank the following: Charlie Bengston for all his work on the initial design and prototyping, Kim Hawkins for her many refinements, Betty Colhoun for her critiques and suggestions throughout the development, all the subsystem engineers for their many ideas and the data technicians who make it so.

10. REFERENCES

1. Computer Sciences Corporation, Hubble Space Telescope Astrometry and Engineering Data Processing System to Post Observation Data Processing System Interface Control Document, ST-ICD-46, February, 1986.
2. Computer Science Corporation, Long Term Trending User's Guide, K. Hawkins and B. Colhoun, October 1992.

N94-23943

HIGH PERFORMANCE, LOW COST, SELF-CONTAINED, MULTIPURPOSE PC BASED GROUND SYSTEMS

Michael Forman, CODE 513
NASA Goddard Space Flight Center
Greenbelt, MD 20771

William Nickum and Gregory Troendly
General Electric Government Services
c/o NASA-GSFC, CODE 602.9
Greenbelt, MD 20771

ABSTRACT: The use of embedded processors greatly enhances the capabilities of personal computers when used for telemetry processing and command control center functions. Parallel architectures based on the use of transputers are shown to be very versatile and reusable, and the synergism between the pc and the embedded processor with transputers results in single unit, low cost workstations of $20 < \text{MIPS} \leq 1000$.

Key Words: workstation, telemetry processing, embedded processor, transputer, parallel processing

1. INTRODUCTION

During the mid 1980's, the flight operations team for the Nimbus 7 satellite developed a ground system concept based on the following goals: low cost, highly enhanced performance, small size, short development cycles, ease of maintenance and use, reusability of software and hardware, and maximal use of commercially available parts and software. The original system used for control of the Nimbus 7 spacecraft was implemented on an IBM 8088 XT with a DOS operating system which housed an embedded processor based on the Motorola 68000 series chip. Since its implementation in 1987, these systems have evolved into Intel based 80486 type PC's with embedded processors which use transputers and modular architectures and operate under SCO UNIX in a multitasking mode. These systems are all data base driven. An individual unit can acquire data at rates in excess of 2 megabits, frame synch it, and perform health and safety analysis, perform all command functions, post pass analysis of all playback data, Level 0 processing, data distribution and archive functions. Section 2 discusses transputers (Ref. 1) and our use of embedded processors in detail and sections 3 and 4 discuss our total integrated ground station.

To date, these systems have been or will be developed and implemented for:

a) The operation of the Nimbus 7 satellite since OCT 1987, at a cost saving of more than \$750,000 per year with less than one day of accumulated down time. Development and testing were completed in less than 2 years.

b) An instrument control center in less than 15 months for the Total Ozone Mapping Spectrometer (TOMS) flown on a Russian Meteor-3 spacecraft on August 15, 1991. To date, more than 99.99% of transmitted data has been captured and distributed.

c) An interface between the NASA communications network and up to ten principal investigators' ground systems in less than 12 months in support of the Global Geospace Satellites' Wind & Polar missions in mid 1991. This was our first on line transputer system.

d) A complete flight operations control center for support of the TOMS Earthprobe mission scheduled for launch in mid 1994.

e) A programmable Telemetry, Command & Archive system for Nimbus 7 operations in less than 8 months. It replaced an equivalent system which occupied 1000 sq. ft. in May 1992.

The transputer is a parallel architecture (Ref. 1) computing device with its own direct communications channels. This allows modular design of the telemetry interface board which results in several tasks to occurring simultaneously such as data ingest and archive, command and verify, and health and safety checks. In practice, we use two or more systems in a cluster mode to perform total spacecraft operations with full redundancy, and due to both hardware and software being identical, systems can be interchanged and perform any required function. In our applications, they are connected via an internal

LAN to a mission planning terminal and a server for data distribution to the Science operations facility. Today's transputers have up to 30 Million Instructions Per Second (MIPS) peak (Ref.1) power and are used for number crunching activities and high speed bit operations, while the Intel based host 80486 computer provides a programming communications interface to the transputer and controls graphic displays and data bases. It is expected that 200 MIPS transputers (Ref 2.) will be available in the near future and this system design will make use of this increased power without the need for redesign.

2. EMBEDDED PROCESSORS

A transputer is the basic building block for a PC based system architecture. The transputer provides both computational power and communication channels which makes it possible to transfer data in a nonbus mode. Transputers feature a built-in hardware scheduler which enables any number of concurrent processes to share the processors time and have well defined growth paths to future technologies. This makes a very powerful and versatile component for Parallel Processing.

Scalable communications band-width is an equally important feature for a parallel architecture. A transputer has four direct DMA link engines which provide highly efficient mechanisms for inter-processor communications and data transfers while avoiding the limitations of a shared memory design. The transputer's DMA links and the ease of integration provides the methodology to support design of functional modular building blocks. In PC based workstations, modularity is the key factor in developing a scalable , cost effective and reliable system as shown in figure 1. Custom modules such as the NASCOM interface, Computation modules, SCSI modules, and Ethernet modules are a few of the modular building blocks utilized in PC based ground stations.

Embedded computers which use Transputers feature a parallel architecture, which in turn provides the ability to expand (or scale) the hardware configuration to suit the application. Parallel Processing relies on the effective utilization of multiple processors for increased performance.

In a PC based ground system, a favorable ratio of computation to communication between processors indicates a good fit for parallel implementations. Problems such as real-time control, monitoring, event

determination, image, and science processing have proven to be good candidates for Parallel Processing.

3. SYSTEM ARCHITECTURE

The current system architectural concepts are:

a.) The host computer's primary function is to provide the programming interface to the embedded system and provide a graphical user interface. The user interface either utilizes a DOS/Windows or UNIX/X-Windows/Motif environment depending on the applications requirements.

b.) The embedded processor provides the systems computational power. Employing 30 MIPS peak (Ref. 1) processing rate per transputer, and capitalizing on the busless point to point communications channels, a network of local task solvers can operate simultaneously. The embedded processor becomes a building block permitting a flexible approach to providing data capture, health and safety analysis, commanding, and science processing all on the same system. As higher speed transputers come on the market, they can replace or compliment slower speed modules.

c.) The use of Commercial Off The Shelf (COTS) hardware and software is maximized and that Industry Standards are utilized for all external interfaces.

A typical system in today's environment consists of an Intel based 80486 host computer with the SCO UNIX Open Desktop operating system and support equipment as diagramed in Figure 2. A special purpose, full length, 32 bit Extended Industry Standard Architecture (EISA) board featuring Bus Mastering Protocol has been designed to provide support of parallel processing. The EISA board provides module sites to support up to ten parallel processing modules as diagramed in Figure 1. The EISA board with the appropriate modules, serves as a front end NASCOM interface with capabilities of hi-speed data capture and archive to disk or tape, frame synchronization and building, cyclic redundancy checks, and packetized data transfer. Other functions such as telemetry processing, calibration, event/mode determination and mode dependent limit checking may be implemented. This provides expandable parallel processing sized to the application. Up to four boards may be installed inside an Intel based 80486 host. Connections are

provided to interface with external based Massively Parallel Processing (MPP) systems (scalable up to 800 Giga-Instructions Per Second (GIPS)) when greater computational capacity is required (Ref. 3). Industry Standard interfaces are utilized and include;

- a) Small Computer Systems Interface (SCSI)
- b) RS-422 and RS-232
- c) Ethernet IEEE 802.3 File Transfer Protocol (FTP), and Transmission Control Protocol (TCP/IP)

4. SYSTEM APPLICATIONS

The system architecture diagramed in figure 2 provides a building block approach to ground system requirements whether it be a stand alone or distributed application. Typically in the stand alone system, one system is used for commanding and another for health and safety analysis. A hot redundant back up for each of the two prime systems prevents a single point of failure and low cost protection for data or commanding loss. Command management, scheduling, and spacecraft software image maintenance is performed on another system with redundant back up. All systems are networked together via ethernet for data distribution, data archive, and strip chart recording. This self contained, integrated system, provides spacecraft command and control capabilities on a desktop and uses standard office power and environmental services.

The use of commercially available software and hardware for data-bases and displays reduces documentation efforts and helps control cost. For example, Case tools are used in the development cycle, word processors are used to develop displays, and FoxPro is used to generated the databases for database driven software. Use of these standard tools reduces learning time both for operators and designers.

The small size does not mean reduced performance. During the pass, real time commanding and command verification based on telemetry returns occur while receiving and processing satellite data. All spacecraft systems are continuously checked and monitored. Icons using color to designate status change are continuously displayed and are selectable for full on

screen viewing by a keystroke. Other features which add to this system performance are;

- a) The capability to do X-Y plots on any telemetry point for trend analysis.
- b) Highlighting status changes due to command or events on any display.
- c) Sub-system sorted chronological by event or mode list.
- d) Use of optical disks for long term data archiving.
- e) Level '0' science processing can be performed post pass or real time depending on the missions needs.

To convert a standalone system to a distributed system, both an ethernet module and compute module are added to the Embedded Controller board. Packets of information are distributed to numerous display terminals using a client-server mode similar to the X-Windows environment. This is well suited for multi-spacecraft operations or applications where multi instrument displays are necessary to support scientist and experimenters.

In summary, the use of personal computers in combination with high speed transputer modules provides a relatively low cost, highly versatile, workstation building block upon which satellite control centers can be built.

5. REFERENCES

1. INMOS. The Transputer Data Book. SGS-Thomson Microelectronics, Second Edition, 1989. INMOS document number: 72 TRN 203 01.
2. INMOS. The T9000 Transputer Products Overview. SGS-Thomson Microelectronics, First Edition, 1991. INMOS document number: 72 TRN 203 01.
3. Parallel Supercomputing. ALTA Technology Corporation. Printed 2/92.

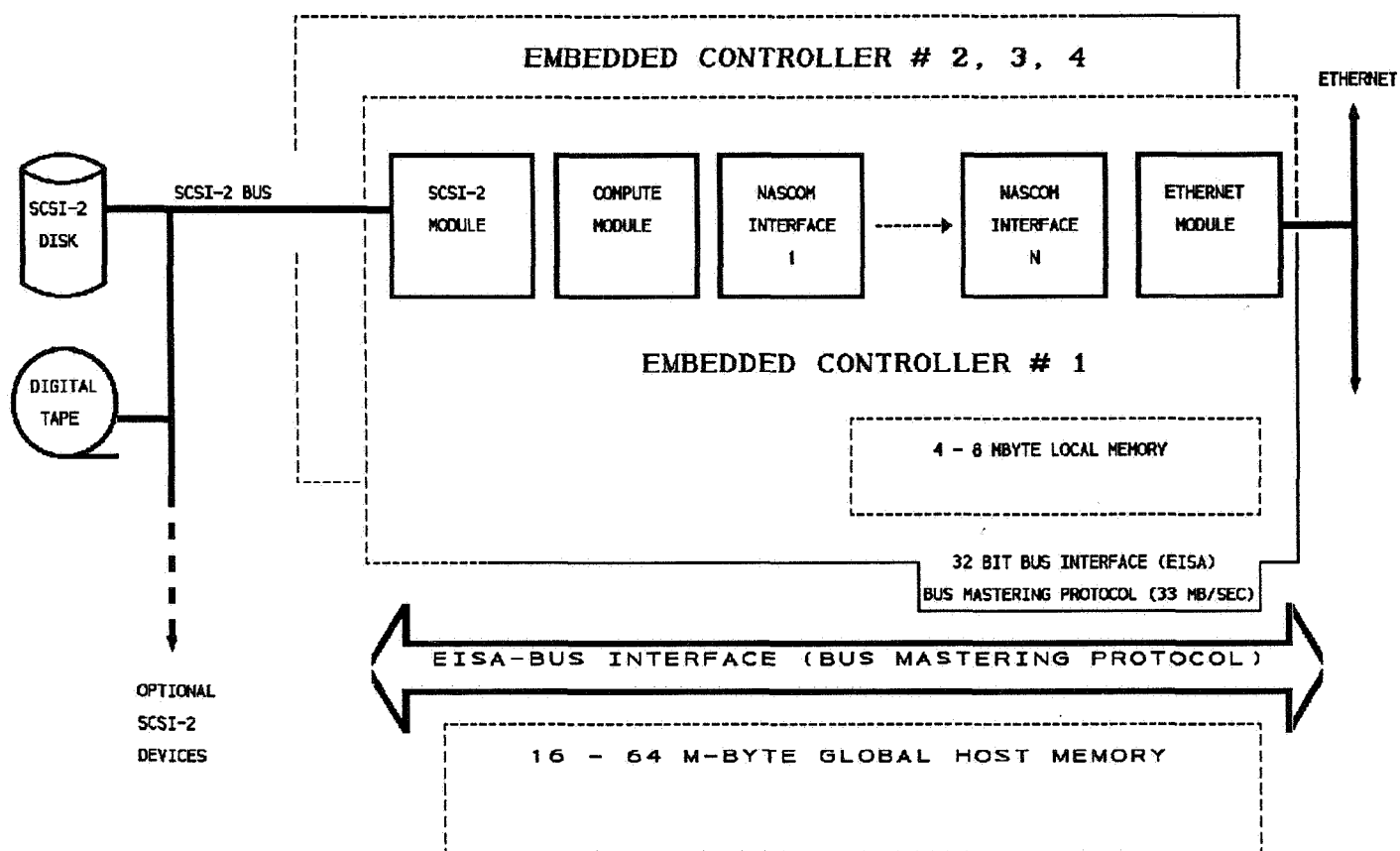
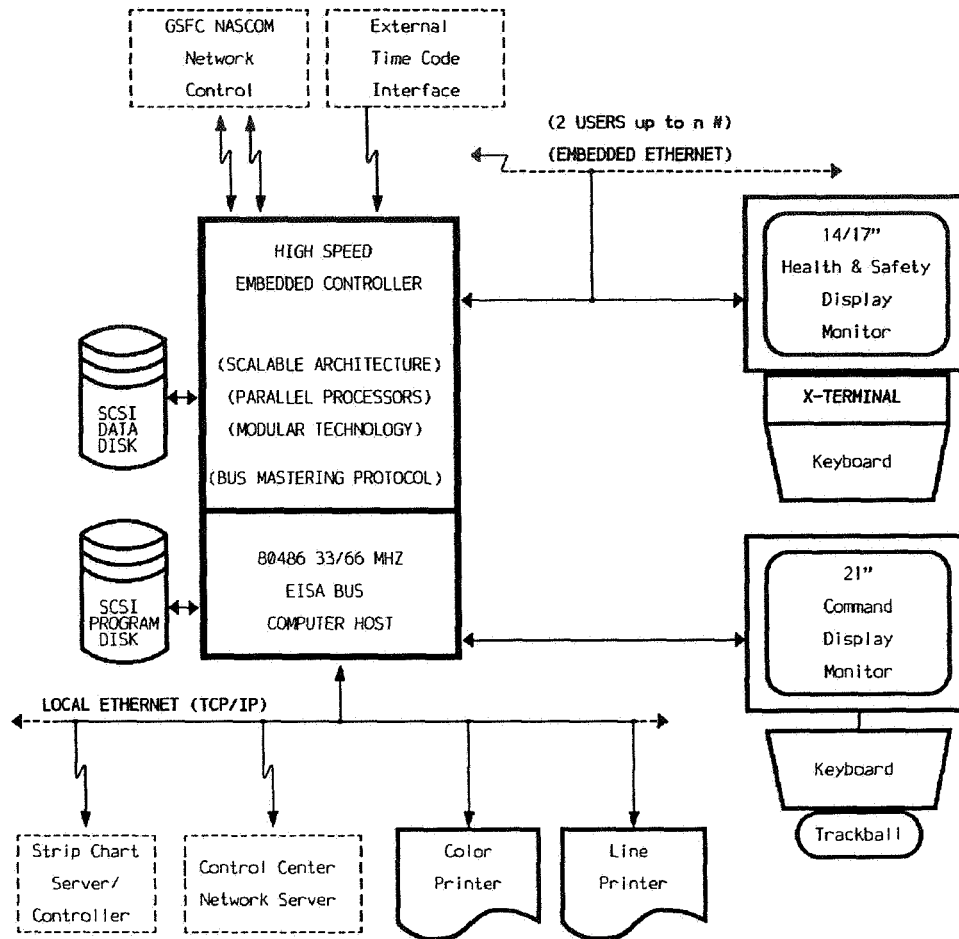


Figure 1. Embedded EISA-BUS Architecture



SCSI - SMALL COMPUTER SYSTEMS INTERFACE EISA - EXTENDED INDUSTRY STANDARD ARCHITECTURE

EMBEDDED CONTROLLER FUNCTIONS

- Expandable Parallel Processing
- Data Ingest: Up to 5.0 Mbit/sec
- Network Deblocking & Data Synchronizing
- High Speed Data Archive Support
- Telemetry Decommutation
- Quality Check of Data
- CRC Encoding / Decoding
- Asynchronous Data Synchronization
- Time Tagging
- Clock Error Determination
- Data Transmit: Up to 1.544 Mbit/sec

HOST FUNCTIONS

- Unix Based Multi-User / Multi-Tasking System
- X-Window Graphical User Interface (GUI)
- Color Video Capture & Print Support
- Strip Chart Recorder Support
- Data Sampling With X-Y Plots
- Trend Analysis
- Telemetry Processing and Display
- Alarm Determination
- Event Determination
- Event Smoothing
- Limit Checking
- Mode Determination
- Gray Code Conversions
- Command Creation, Edit, Validation
- Command Verify
- System Control
- Data Base Management

Figure 2. System Design Concept

EXPERT DIAGNOSTICS SYSTEM AS A PART OF ANALYSIS SOFTWARE FOR POWER MISSION OPERATIONS

Jennifer A. Harris and Khosrow A. Bahrami

N94-23944

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA

ABSTRACT

The operation of interplanetary spacecraft at JPL has become an increasingly complex activity. This complexity is due to advanced spacecraft designs and ambitious mission objectives which lead to operations requirements that are more demanding than those of any previous mission.

For this reason, several productivity enhancement measures are underway at JPL within mission operations, particularly in the spacecraft analysis area. These measures aimed at spacecraft analysis include: the development of a multimission, multisubsystem operations environment, the introduction of automated tools into this environment, and the development of an expert diagnostics system.

This paper discusses an effort to integrate the above mentioned productivity enhancement measures. A prototype was developed that integrates an expert diagnostics system into a multimission, multisubsystem operations environment using the Galileo Power / Pyro Subsystem as a testbed. This prototype will be discussed in addition to background information associated with it.

1. INTRODUCTION

The operation of interplanetary spacecraft at JPL has become an increasingly complex activity. This complexity is due to advanced spacecraft designs and ambitious

mission objectives which lead to operations requirements that are more demanding than those of any previous mission. JPL is now at a point where the current mission operations infrastructure is not sufficient to support future mission operations requirements unless this support is provided through increased staffing. Because of the cost constraints now being placed on spacecraft operations by NASA, staffing increases are not an option and an improvement over the current mission operations infrastructure is needed.

In order to address this problem, several software development efforts are underway at JPL. One such effort is the Engineering Analysis Subsystem Environment (EASE) prototype. EASE is a collection of software programs providing a framework within which spacecraft engineering analysis can be done. This multimission, multisubsystem environment provides tools and a friendly graphical user interface to support spacecraft engineering analysis activities such as telemetry monitoring and performance predictions.

Another effort that has been ongoing in order to address the mission operations problem is the Spacecraft Health Automated Reasoning Prototype (SHARP). SHARP is an expert system that contains subsystem specific knowledge which is used to provide diagnostics about that subsystem. SHARP was developed to assist the engineering analysts in diagnosing anomalous behavior on the spacecraft

and recommending actions to take in these situations.

Because the operations environment provided by EASE and the diagnostics capabilities provided by SHARP compliment each other, the two prototypes were integrated to provide a more complete operations environment for spacecraft engineering analysis. This integrated system uses the Galileo Power / Pyro subsystem as its prototype subsystem.

Following is a discussion of the EASE / SHARP prototype in terms of its integration and rules. Background information concerning the JPL mission operations process, EASE, and SHARP is also provided.

2. MISSION OPERATIONS

Spacecraft operations at JPL involves several steps. It includes developing sequences to send commands to the spacecraft, receiving telemetry from

the spacecraft, and analyzing this telemetry to characterize the spacecraft state. Figure 1 shows a simplified mission operations process.

The uplink process is shown across the top of the diagram. It begins with science and engineering requests being mapped with the Deep Space Network (DSN) coverage and other requirements to create a mission plan. The mission plan is a time ordered set of activities. This set of activities is then expanded into a time ordered set of commands through preliminary and final sequence generation. The validation of sequences and resolution of conflicts is an interactive process between the sequence and spacecraft analysis elements of mission operations.

The downlink process begins when the spacecraft sends telemetry via the DSN to the JPL Space Flight Operations Center (SFOC). This data is decommutated, stored in a database, and

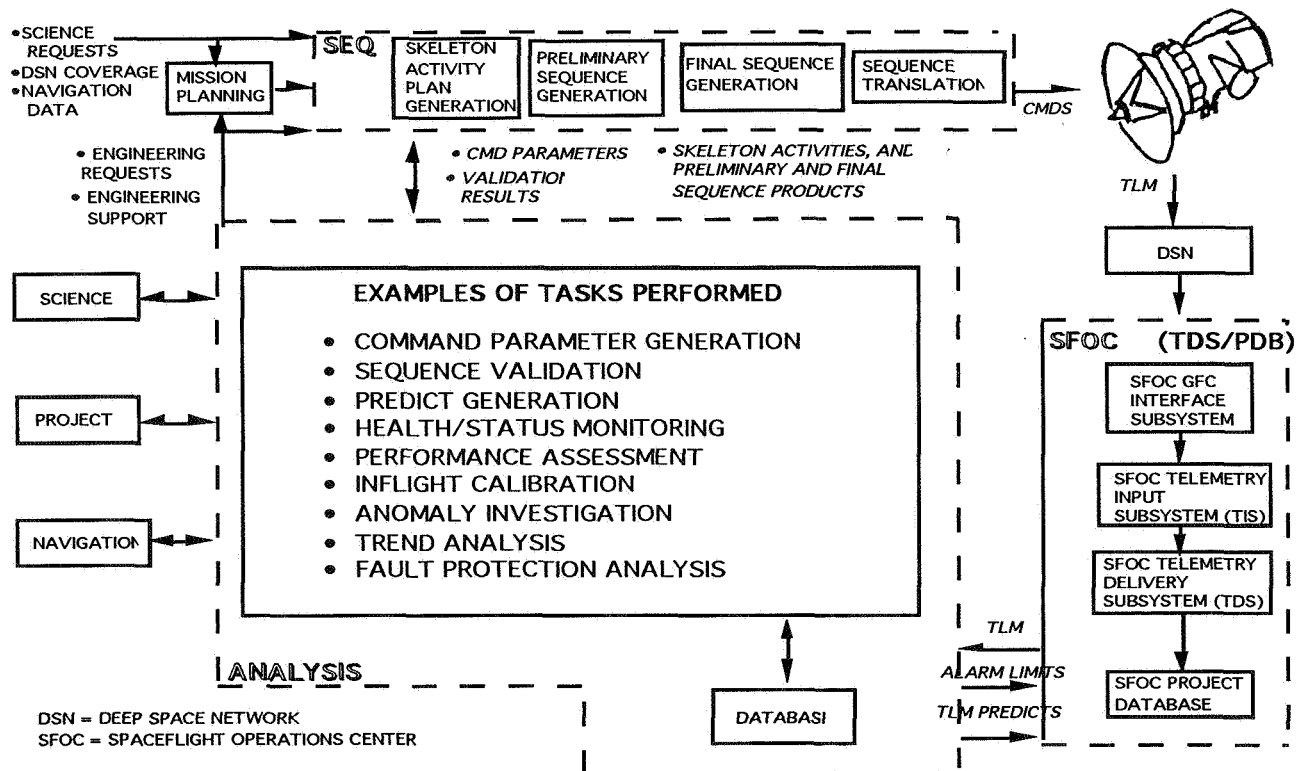


FIGURE 1. A typical Spacecraft Operations Process

delivered to the engineering analysts.

The engineering analysis downlink portion includes performance monitoring, trend analysis, anomaly investigation, performance assessment, and fault protection analysis. Uplink support functions carried out by engineering analysis include command parameter generation, sequence validation, and predict generation. [Ref. 1 - 3].

The EASE / SHARP prototype addresses the engineering analysis portion of the operations process. It provides a software environment within which engineering analysis tasks can be carried out more efficiently through automation, software tools, and expert systems diagnosis.

3. THE ENGINEERING ANALYSIS SUBSYSTEM ENVIRONMENT (EASE)

The EASE system is a modular multimission, multisubsystem software environment. For a particular mission, various mission specific subsystem analysis modules (SAMs) are added to the environment to provide subsystem specific telemetry processing. Figure 2a shows the environment when

applied to a single mission and populated with SAMs. In a typical case, the SAMs are as follows:

- Attitude and Articulation Control Subsystem (AACS)
- Command and Data Handling Subsystem (CDS)
- Electrical Power Subsystem (EPS)
- Propulsion Subsystem (PROP)
- Systems Subsystem (SYS)
- Telecommunications Subsystem (TELECOM)
- Temperature Subsystem (TEMP)

When two or more single mission applications are interconnected through a multimission graphical user interface (MMGUI), it forms a multimission analysis system as shown in Figure 2b. As can be seen from the diagram, the modular EASE architecture facilitates the integration of new SAMs for a specific mission in addition to applications for different missions.

The functionality of the EASE environment is provided through common interface formats, common display windows, development tools,

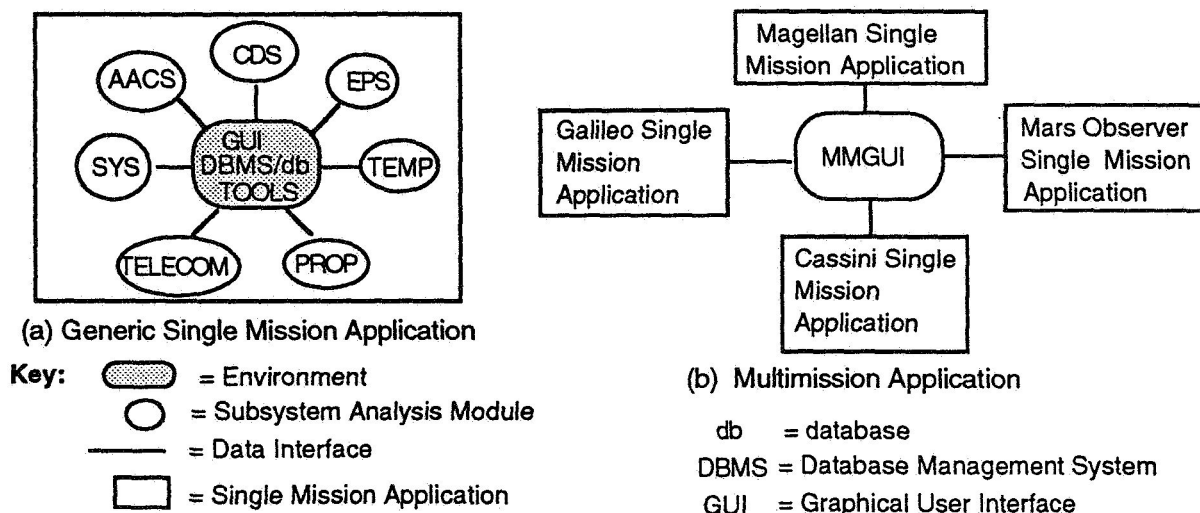


Figure 2. The EASE Architecture

analysis tools, and a Database Management System (DBMS). By using the environment as a framework, SAMs can be integrated efficiently. Also the DBMS and tools provide opportunity for automated data analysis and manipulation.

The EASE prototype currently includes several subsystem modules such as: GLL PPS and CDS, Voyager (VGR) EPS, and Mars Observer (MO) PPS. The most developed of these subsystems is the GLL PPS.

4. THE SPACECRAFT HEALTH AUTOMATED REASONING PROTOTYPE (SHARP)

SHARP is a diagnostics system that has been described in depth in a previous report [Ref. 5]. It is written in LISP and has been developed for the VGR and Magellan (MGN) TELECOM subsystems. The VGR and MGN versions of SHARP are stand alone implementations that contain a diagnostics kernel, VGR and MGN specific TELECOM knowledge, and diagnostics displays.

SHARP for the GLL PPS subsystem implementation contains just the SHARP diagnostic kernel and GLL PPS specific subsystem knowledge.

5. THE EASE / SHARP SYSTEM

The primary motivation for integrating SHARP into the EASE framework is to enhance the EASE operations environment developed for monitoring and analyzing the spacecraft data by adding capabilities for diagnosing the state of the spacecraft. Below is a discussion of the integration process of SHARP into EASE and the rules currently included in the SHARP diagnostics module.

5.1 Integration

The first step in the integration process was to separate the SHARP

kernel from the rest of the SHARP system. An interface between the C and LISP programming languages is used to pass data from EASE to SHARP. SHARP uses this data to determine alarms and anomalies on the spacecraft and reports these back to EASE. A LISP to C programming language interface is used to send alarm and diagnostics messages from SHARP to EASE.

These alarm and diagnostics messages contain a substantial amount of subsystem specific information. For channels in alarm, the messages contain the mission name, channel type, channel number, time, alarm limits, alarm state (high hard, low soft etc.), a title message, summary message, and a message key which points to a detailed message about the problem. The diagnostics messages contain similar information with most of the subsystem specific knowledge being included textually in the title, summary, and detailed messages.

5.2 Rules

The set of rules the prototype uses for testing is for diagnosis of a GLL PPS bus undervoltage. Although this is a rare anomaly, the reasoning process in determining the cause of a bus undervoltage is well understood providing a useful basis for developing and testing this prototype. The GLL PPS power bus is tightly regulated between 30.1 and 31.5 volts. If the voltage on the bus goes under the allowable limit of 30.1 volts, the bus undervoltage rules within SHARP will trigger. SHARP uses all related telemetry in order to determine the cause of the anomaly. Possible causes for a bus undervoltage anomaly include a failure in a shunt regulator stage, a dc bus short, an ac bus short, or an ac inverter failure.

Another set of rules in development for SHARP is for determining load status (power consumption value of loads). The GLL spacecraft has 121 total

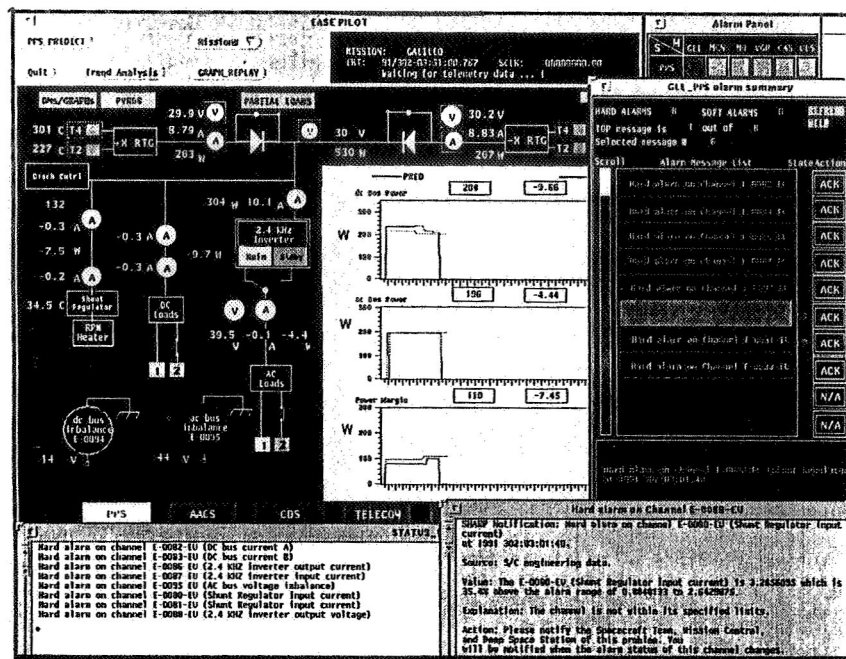


FIGURE 3. The EASE / SHARP Display Screen

dc and ac loads. Power status telemetry that gives visibility into these load states consists of the bus and ac inverter output voltages, and the ac inverter output, shunt regulator, and dc bus currents. Additional information such as thermal and command data must be used by SHARP in order to infer the state of each of the 121 spacecraft loads. Because of telemetry and processing constraints on the spacecraft, determining the state of spacecraft loads on GLL must be done by the Ground System.

5.3 Displays

The displays for the EASE / SHARP prototype are developed within the EASE environment. The "LISP" to "C" interface is used to pass data from SHARP to the EASE display module.

In designing the displays, concerns about the overload of information whenever an anomaly or several alarms occur was addressed. The display design chosen for the EASE / SHARP prototype uses the concept of "layering" information to deal with this problem. "Layering", as used here, means presenting summary information to the user at all times. More detailed information concerning an alarm or diagnosis can be accessed by the user simply by clicking the

mouse on the summary information. In this way, the analyst is aware of all alarms and diagnoses and can access any information about them, but is not overwhelmed with large amounts of detailed data.

Figure 3 shows the GLL PPS block diagram shortly after the occurrence of a simulated bus undervoltage anomaly. The top three layers of display windows are shown on the right side. Layer 1 is the alarm panel, layer 2 the alarm summary, and layer 3 the detailed alarm message. The operator may acknowledge or ignore alarms that are listed on the alarm summary table.

6. CONCLUSION

Through the development of prototypes such as the EASE / SHARP prototype, much can be learned about using software to improve the mission operations process at JPL. This particular prototype has given valuable insight into the advantages of using a modular, multimission, multisubsystem architecture as a framework for integrating analysis modules such as SHARP. Data was also obtained concerning how to display large amounts of information, and the importance of operations considerations in spacecraft design.

The modular, multimission, multisubsystem architecture provided by the EASE environment was found to be a stable, efficient way of incorporating different capabilities into one system. Interface code from other modules within EASE could be modified and used to integrate SHARP and the new displays. The idea of using a common environment to provide displays, interfaces between modules, and common tools made the integration of SHARP into EASE straightforward.

Because of the additional analysis information that was added to the EASE system through the integration of SHARP, the display of information became a concern. A "layering" approach was selected and was useful in showing the operator summary information, and allowing for detailed information to be accessed.

A final point that was learned in this process is the possible implications of "operationally blind" spacecraft design. Much time was spent in knowledge engineering and software development for the load status rules. This would not be necessary if appropriate telemetry was provided from the spacecraft. The Cassini project has realized this and has included load current telemetry for all Cassini loads. This problem is not specific to the power subsystem, however, and early operations involvement in spacecraft design is a necessity at both the subsystem and system level in order to avoid problems such as this.

The EASE / SHARP prototype has provided useful information in exploring the use of advanced software for mission operations. The purpose of these software packages is to provide functionality and assistance to the analyst so that fewer analysts will be needed for mission operations and cost reductions in this area will be possible.

7. ACKNOWLEDGEMENTS

The work described in this paper was carried out at the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

The authors hereby acknowledge the valuable contribution of the various EASE and SHARP team members, particularly, K. Hioe, E. Hsu, E. Imlay, M. James, M. Lei, R. Martin, S. Mikes, and A. Moncada.

8. REFERENCES

- (1) K. A. Bahrami and K.L. Atkins, "Automated Workstation for the Operation of Spacecraft Engineering Subsystems," Proc. 23rd IECEC, July 31-Aug. 5, 1988, Denver, Colorado; New York, ASME, pp. 367-375, 1988.
- (2) K. A. Bahrami, "A New Environment for Multiple Spacecraft Power Subsystem Mission Operations", Proc. 25th IECEC, August 12-17, 1990. Vol 1, pp. 345-352.
- (3) K. A. Bahrami, et al., "An Automated Environment for multiple Spacecraft Engineering Subsystem Mission Operations", Proc. AIAA/NASA Second International Symposium on Space Information Systems", 17-19 September 1990, Pasadena, CA. Vol 2 pp 991-1005.
- (4) K. A. Bahrami and J. A. Harris, "An Advanced Environment for Spacecraft Engineering Subsystem Mission Operations", Proc. 27th IECEC, August 3-7, 1992, San Diego, CA.
- (5) R. G. Martin, et al., "A Report on SHARP and the Voyager Neptune Encounter", JPL Publication 90-21.

GenSAA: A Tool For Advancing Satellite Monitoring With Graphical Expert Systems

Peter M. Hughes

Software and Automation Systems Branch (Code 522)
Mission Operations and Data Systems Directorate
NASA/Goddard Space Flight Center
Greenbelt, Maryland 20771

Edward C. Luczak

Computer Sciences Corporation
4600 Powder Mill Road
Beltsville, Maryland 20705

N94-23945

Abstract

During numerous contacts with a satellite each day, spacecraft analysts must closely monitor real time data watching for combinations of telemetry parameter values, trends, and other indications that may signify a problem or failure. As satellites become more complex and the number of data items increases, this task is becoming increasingly difficult for humans to perform at acceptable performance levels. At the NASA Goddard Space Flight Center, fault-isolation expert systems have been developed to support data monitoring and fault detection tasks in satellite control centers. Based on the lessons learned during these initial efforts in expert system automation, a new domain-specific expert system development tool named the Generic Spacecraft Analyst Assistant (GenSAA), is being developed to facilitate the rapid development and reuse of real-time expert systems to serve as fault-isolation assistants for spacecraft analysts. Although initially domain-specific in nature, this powerful tool will support the development of highly graphical expert systems for data monitoring purposes throughout the space and commercial industry.

Key Words: expert systems, mission control, X-windows, automation, real-time, data monitoring.

1. Introduction

NASA's Earth-orbiting scientific satellites are becoming increasingly sophisticated. They are operated by highly trained personnel called Flight Operations Analysts (FOAs) in the satellite's control center. The FOA's are responsible for the proper command, control, health, and safety of the satellite [Ref. 3].

Goddard's satellite control centers operate round-the-clock throughout the lifetime of the spacecraft. There are typically multiple real-time communications events daily with each satellite. During these events, the FOAs must:

- establish and maintain the telecommunications link with the spacecraft,
- monitor the spacecraft's health and safety,
- send commands or command loads to the satellite for on-board execution,
- oversee the transfer of the scientific data from the on-board tape recorders to ground systems for processing and analysis,
- manage spacecraft resources (e.g., on-board memory, power, and tape recorders).

To accomplish these activities, the analyst must thoroughly understand the operation of the spacecraft and ground systems and continuously monitor the current state of operations as indicated by telemetry parameters displayed on the Payload Operations Control

Center (POCC) consoles. During an event, the analyst typically monitors hundreds of telemetry parameter values on multiple display pages that may be updated several times per second. Such large volumes of low-level information can overwhelm analysts and disrupt their ability to identify and resolve conflicting constraints. They may soon be unable to consistently monitor all of the information available. The need to automate some data monitoring functions is apparent.

Expert system technology is proving to be effective in automating some spacecraft monitoring functions. This paper first summarizes CLEAR, the first real-time spacecraft monitoring expert system deployed at GSFC. The paper then reviews several lessons learned from CLEAR and other monitoring and fault isolation expert system projects undertaken at GSFC, thereby establishing the foundation of a domain-specific expert system development tool called the Generic Spacecraft Analyst Assistant (GenSAA). This new tool will be introduced followed by a discussion of its capabilities, architecture and benefits, and the approach for adapting this tool to other domains.

2. The CLEAR Expert System

The Communications Link Expert Assistance Resource (CLEAR) is the first operational expert system at GSFC that automates one of the spacecraft analyst's tasks [Ref. 2]. It is a fault-isolation expert system that supports real-time operations in the POCC for the Cosmic Background Explorer (COBE) mission. CLEAR monitors the communications link between COBE and the Tracking and Data Relay Satellite (TDRS), alerts the analyst to any problems, and offers advice on how to correct them.

CLEAR is a forward chaining, rule-based system that operates in the COBE POCC. It monitors over 100 real-time performance parameters that represent the condition and operation of the spacecraft's communications with the relay satellite. Using this information, together with knowledge of TDRS operations, COBE's on-board communications system and the expected configuration of the scheduled event, CLEAR accurately portrays the status of the communications link.

Textual and graphical information about the condition of the COBE/TDRS/ground communications links is displayed in a tiled-window format. A graphics window displays the elements of the communications network from the COBE Spacecraft to the POCC; green lines represent healthy links between elements. When the performance parameters indicate that a communications link or processing system is degrading or down, the

associated line or icon turns yellow or red, respectively. The display enables analysts to assess the current status of the communications event in a quick glance.

When CLEAR isolates a problem, a short description of the problem is displayed in the "problems" window. If multiple problems are found, the problem descriptions are ranked and displayed in descending order of criticality. CLEAR suggests analyst actions to correct the problem; however, the system does not take any corrective action itself.

To further assist the analyst and to provide support for its advice, the CLEAR system provides an explanation facility. When the analyst selects a problem displayed in the problems window, CLEAR provides a detailed explanation of why the expert system believes that the problem exists.

CLEAR has approximately 165 rules and isolates approximately 75 different problems. The types of problems include: non-reception of data within the control center (system or communication problems, or data reporting not activated); misconfigurations between the COBE POCC and the TDRS ground station (coherency/non-coherency, doppler compensation on/off, power mode, actual TDRS in use, antennae configurations); discrepancies in telemetry rate or format; inactive or non-locked links; and degrading or critical signal strength situations [Ref.6].

CLEAR operates on any of the seven PC/AT-class workstations that are used for console operations in the POCC. It is written in the 'C' language and uses the 'C' Language Integrated Production System (CLIPS) and a custom-developed graphics library.

The CLEAR Expert System has supported the COBE Flight Operations Team since launch in November 1989. It is used to monitor nearly all of the TDRS supports (COBE periodically communicates directly to the Wallops ground station) and is regarded as the fault-isolation "expert" for the COBE/TDRS telecommunications link. CLEAR represents a successful attempt to automate a control center function using an expert system. It has been adapted for the Gamma Ray Observatory and was utilized during early orbit.

2.1 Lessons Learned from CLEAR

Several important lessons have been learned from the experience gained in developing and operating CLEAR [Ref. 2]. Key lessons have also been learned from other monitoring and fault isolation expert systems developed recently at GSFC, including the Ranging Equipment Diagnostic Expert System (REDEX) [Ref. 5]. The following is a subset of the lessons learned from these projects that have strongly motivated and influenced the development of GenSAA. [See Reference 2 for a complete listing and detailed description of each.]

- *Production rules effectively represent analysts' knowledge for automating fault-isolation in spacecraft operation.* The rule-based method of knowledge representation has proven to be quite powerful for fault-isolation in spacecraft operations. Production rules provide a direct method of encoding the fault-isolation knowledge of spacecraft analysts because the if-then structure closely parallels the stimulus-response behavior

that they develop through extensive training.

- *Allow the domain expert to participate in the rule formation.* The participation of the domain expert in knowledge acquisition and translation has many advantages: it can reduce knowledge translation time and errors; facilitate knowledge verification and validation; and improve user confidence in the system thereby better ensuring acceptance.

- *Expect to maintain/fine-tune the expert system after it becomes operational.* Although the operational characteristics of a spacecraft can be fairly accurately predicted, actual behavior can vary. The use of rule constructs facilitates changes necessary to refine the system quickly and easily.

- *Don't underestimate the effort to necessary to integrate the expert system with the data acquisition and user-interface elements.*

- *Provide an inviting, intuitive, and user-configurable user-interface.* The human-computer interface is frequently the most underdeveloped component of an expert system. Often, it can be the factor that determines how well the system will be accepted and trusted by the end-user.

- *Capitalize on the expressive power of graphics.* Graphics can greatly enhance the visual communications of a system; capitalize on their expressive power to provide system output that can be assimilated quickly and accurately.

- *Simplify the operation of the system as much as possible by:*

- *Reducing user input to a minimum.*
- *Using hypergraphic techniques to allow quick and intuitive access to information and navigation through the system.*

3. GenSAA

GenSAA is an advanced tool that will enable spacecraft analysts to rapidly build simple yet highly graphical expert systems that are capable of performing real-time spacecraft monitoring and fault isolation functions. These expert systems will assist the flight operations team in spacecraft control centers.

Expert systems developed with GenSAA will have the following characteristics:

- Easily created and modified— The process for developing specific expert systems using GenSAA will be straightforward enough that it can be performed by trained spacecraft analysts on the flight operations team. No compilation step is necessary before executing the expert system.

- Rule-based— GenSAA will support the use of rules to represent spacecraft and payload monitoring and fault isolation knowledge. Rule-based representations are easily learned and can be used to describe many of the reasoning processes used by spacecraft analysts.

- Highly graphical— The GenSAA operational user interface will support both textual and graphical presentations of health and status information and fault isolation conclusions. GenSAA user interfaces are built with the GenSAA WorkBench which uses the X-window toolkit and the Motif widget set. Hyperlink techniques

will be supported to simplify navigation between GenSAA windows.

- Transparently interfaced with the data source—Initially, GenSAA will be used to create expert systems that will support analysts in spacecraft control centers that use the new Transportable Payload Operations Control Center (TPOCC) architecture. TPOCC is a new Unix-based control center system architecture that will be used for many new spacecraft missions at GSFC. GenSAA will also be adaptable to support non-TPOCC data interfaces.

- Real time—GenSAA expert systems will be driven by real time spacecraft telemetry that indicate the current status of the spacecraft and its operation.

GenSAA is being developed as a generic tool to support the development of expert systems for TPOCC-based control centers supporting missions in the Small Explorer (SMEX) and International Solar-Terrestrial Physics (ISTP) programs. The initial use of GenSAA is targeted for the SAMPEX, FAST and the WIND/POLAR missions.

3.1 GenSAA Architecture

The GenSAA architecture is based on distributed processing and a variety of emerging industry standards. It employs the client/server model of distributed processing, the Network File System (NFS) protocol for transparent network access to files, and the X Window System (X.11) with the Motif library and window manager for the graphical operator interface. It is designed to operate in a TPOCC network of computers which consists of a small set of specialized front-end processors and Unix-based workstations on an Ethernet network using the TCP/IP network protocol.

3.2 The GenSAA Workbench

The GenSAA Workbench is composed of three utilities that enable a spacecraft analyst to create a GenSAA expert system which performs real-time monitoring and fault isolation functions in a TPOCC spacecraft control

center.

The GenSAA Workbench will operate in an off-line mode on a Unix workstation. A GenSAA expert system is created by defining the expert system's runtime specifications using the GenSAA Workbench. Figure 1 illustrates that these specifications, called Reusable Application Components, together with the GenSAA Runtime Components, compose a GenSAA expert system. The GenSAA Workbench utilities are as follows:

- Data Manager—This utility is used to create the Data Interface Specification for a GenSAA expert system. The Data Interface Specification defines four types of data that are used by the GenSAA expert system during real-time operations:

- Mission data—Mission data variables represent real-time status of the monitored spacecraft and related ground support systems. (Mission variables are sometimes called telemetry mnemonics.) Values for these variables are received and updated during spacecraft operation periods from the TPOCC Data Server process, which is part of the TPOCC software. Using the Data Manager, the GenSAA Workbench user selects the mission variables needed for the expert system being created from a list of all the mission variables available from the TPOCC Data Server. Values for only those variables selected will be received by the expert system during run-time.

- User-defined data—User-defined data variables represent expected operating modes and equipment configurations. For example, a user-defined data variable might represent the setting of a switch that determines which of two redundant components is to be used. Values for these variables are entered by the spacecraft analyst during spacecraft operation periods.

- Inferred data—Inferred data variables represent conclusions inferred by rules in the rule base. For example, an inferred data variable might represent the health or fault status of a component in a spacecraft subsystem. Values are assigned to these variables by

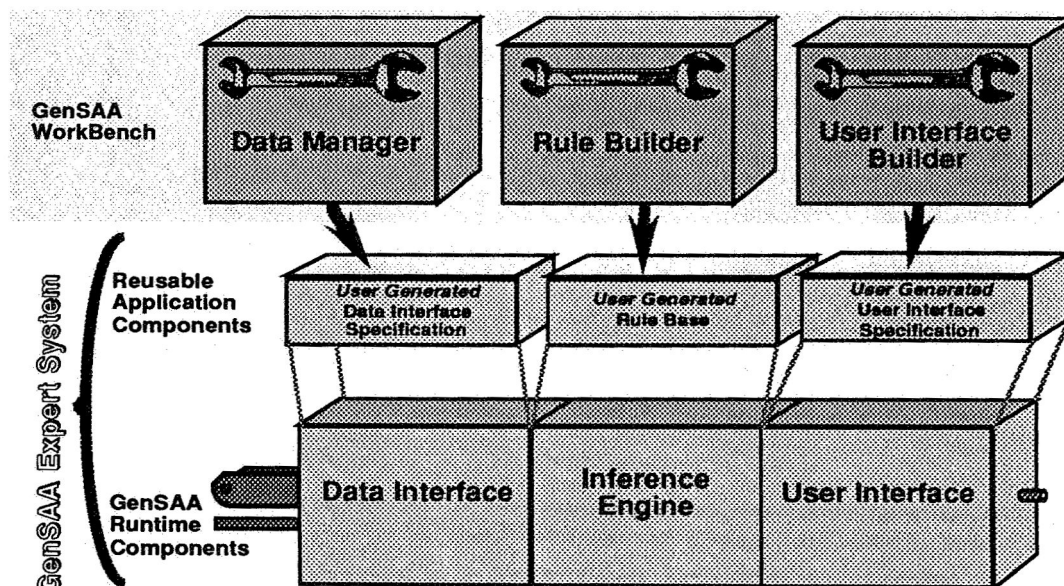


Figure 1. The Elements of GenSAA

actions executed in the "then" part of rules that fire.

– *Externally Generated GenSAA (EGG) data*– EGG data consists of Inferred and User-Defined data which is available from other GenSAA expert systems. Conversely, a developer of a GenSAA expert system can identify Inferred and User-Defined data as "public" causing them to be routed to the GenSAA Data Server which distributes them to any process requesting them. For example, one GenSAA expert system may require information about the status of a subsystem which is being monitored by another GenSAA expert system. Such inter-expert system communication is conducted through EGG data.

• **Rule Builder**– This utility is used to create the rule base for a GenSAA expert system. The rule base is a set of expert system rules in "condition-action" ("if - then") format that may infer new facts based on currently asserted facts. The inference engine manages the matching and firing of rules in the rule base during execution of the GenSAA expert system.

During run-time, if all the conditions of a rule are satisfied, then the rule fires and all its actions are performed. Conditions can be constructed using the mission, user-defined, and inferred data variables specified with the Data Manager. Actions may include: asserting/retracting an inferred fact, enabling/disabling a rule or ruleset, performing a mathematical calculation, and displaying text messages on the user interface. Templates are provided for specifying conditions and actions thereby allowing a user to build rules quickly using drop-and-drag techniques.

• **User Interface Builder**– This utility is used to create the User Interface Specification for a GenSAA expert system. The User Interface Specification defines the user interface windows and the layout and behavior of the graphical objects that comprise the operational user interface of the expert system.

The Workbench user can use a variety of X-toolkit and Motif widgets, including pushbuttons, option menus, scrolling text lists, user-created graphical icons, and data-driven objects such as meters, gauges, and miniature stripcharts. The designer constructs a user interface by selecting graphical objects from a palette or drawing them with the graphics tools provided and positioning them where desired. Dynamic lines can be drawn to create animated schematic diagrams. The Workbench user can associate each graphical object with a mission, user-defined, inferred data, or EGG variable, and specify how changes in the value of the variable will affect the presentation of the item. Characteristics of a graphical object's behavior that can change based on the value of its associated data variable include its color, the icon displayed, and the position of the dynamic portion of a data-driven object. Simple drawing editors are provided to allow the creation of new graphical icons. Any graphical object can also be specified to be a hyperlink button; clicking on such a button during run-time can cause one or more windows to be displayed.

The GenSAA Workbench utilities are highly interoperable and use a graphical, direct-manipulation method of interaction (commonly referred as "point-and-select" or "drag-and-drop") to facilitate use. For example, when using the Data Manager, the user may

select a given mission mnemonic to be included in the Data Interface Specification. Later, when using the Rule Builder, the user can drag the mnemonic from the Data Manager into a condition of a rule. Similarly, when using the User Interface Builder, the user can drag a GenSAA data variable onto a graphical item in the user interface to associate the variable with the graphical object. This pointing technique prevents keyboard mistakes and is faster than typing.

GenSAA expert system components will be placed in a library so that they can be reused in creating the specifications for new GenSAA expert systems. Operations like cut and paste will be available to allow portions of previously created specifications to be used in constructing a new expert system.

3.3 GenSAA Runtime Framework

The GenSAA Runtime Framework provides the basic operational environment for a GenSAA expert system. The elements of the GenSAA Runtime Framework are called the GenSAA Runtime Components; they are used without change in each GenSAA expert system. They control the operation of a GenSAA expert system during its execution in a TPOCC control center. They read the Data Interface Specification, Rule Base, and User Interface Specification files to determine the specific behavior of the GenSAA expert system. The GenSAA Runtime Framework is implemented as a pair of Unix processes that communicate with one another via message queues. Their functions are as follows:

• **User Interface Process**– This component manages the operation of the expert system's user interface. It handles the user input, manages the display of windows (that may contain both text and graphics) and updates the visual attributes of the graphical objects. The user interface windows, data-driven objects, and interaction objects are defined in the User Interface Specification that was generated by the GenSAA User Interface Builder.

• **Data Interface Subsystem**– This sub-element of the User Interface Process requests mission data from the TPOCC Data Server, as specified in the Data Interface Specification. It formats the real-time data it receives and distributes it to the Inference Engine and User Interface components.

• **Inference Engine Process**– This component manages the firing of rules in the rule base. A rule is fired when all its conditions are satisfied; the conditions will often involve the current values of mission, user-defined, EGG, and inferred data variables. Inferred facts and messages may be sent to the User Interface component and displayed to the user as defined in the User Interface Specification. NASA's 'C' Language Integrated Production System (CLIPS) inference engine forms the core of this component.

The user interface of a GenSAA expert system will typically include color schematics and animated data-driven objects (such as rotating meters, sliding bar graphs, and toggle switches) that graphically display the dynamic values of telemetry data, user-defined data, and inferred conditions. The user interface will also typically contain hypergraphic links to make it easy for the spacecraft analyst to quickly display graphics windows.

A GenSAA expert system will execute on a Unix workstation in the control center. A dedicated Unix workstation is not required, i.e., a GenSAA expert system can execute on the same workstation as other TPOCC processes. However, to avoid potential performance impacts, the initial GenSAA expert system will reside on a dedicated Unix workstation connected to the TPOCC Local Area Network.

During operation, GenSAA expert systems will interface to the TPOCC software via the TPOCC Data Server process. This interface will use the standard external interface conventions defined by the TPOCC Project. For example, a GenSAA expert system simply submits a request to the TPOCC Data Server process for the mission data that was specified in the expert system. No additional data or commands are sent to any of the TPOCC processes.

3.4 Implementation

GenSAA is implemented in C++ using an object-oriented design. This approach was selected because of the following four benefits: First, it allows the reuse of an existing class library developed at GSFC (Code 522) for the rapid development of software. Second, it promotes modularity and ease integration of the software components that will comprise GenSAA. Third, it will allow the core modules of GenSAA to be implemented so that the system can be extended for future missions or industrial use without major design changes or extensive recoding. Fourth, the GenSAA development team is optimistic that the object-oriented approach will facilitate maintenance of this system.

3.5 Cooperating GenSAA Expert Systems

GenSAA expert systems are intended to be relatively simple expert systems with small rule bases that are typically developed by a single analyst. A typical GenSAA expert system might monitor and isolate faults for one subsystem onboard a spacecraft. To handle more complex monitoring situations, involving, for example, several spacecraft subsystems, multiple GenSAA expert systems can be built each responsible for a discrete subsystem or function. During operation, these expert systems would execute concurrently and could share key conclusions with one another using a "publish-and-subscribe" model of communicating.

To perform the publish-and-subscribe method of information sharing, a fourth component of the GenSAA Runtime Framework, the GenSAA Data Server, is used to serve as a central repository to which GenSAA expert systems can "publish" information and from which other "subscribing" GenSAA expert systems can receive the information when published. As shown in Figure 2, the GenSAA Data Server is a Unix process that can receive a real-time stream of "public" user-defined and inferred data variable updates from any GenSAA expert system. The GenSAA Data Server distributes the data to any GenSAA expert system that has requested it. A given expert system only receives those variables it specifically requested (subscribed). The data received by a GenSAA expert system from the GenSAA Data Server is called Externally Generated GenSAA (EGG) data. A GenSAA expert system receives EGG data via its Data Interface component in exactly the same way as it receives

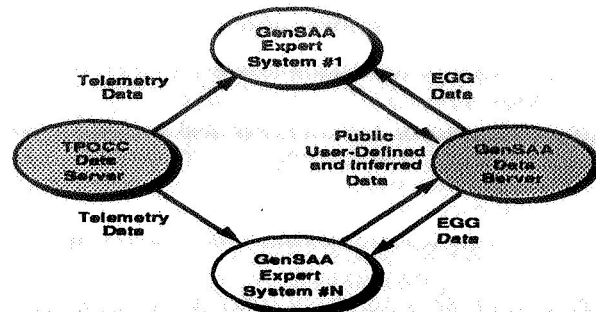


Fig 2. Sharing data among multiple GenSAA E.S.'s

mission data from the TPOCC Data Server.

Within a GenSAA expert system, EGG data can be used in the conditions of rules, and can be associated with display items in exactly the same way as mission, user-defined, and inferred data. The Workbench supports the selection of EGG data as a fourth variable type. The Workbench also allows any local user-defined or inferred data to be specified as "public", to cause it to be sent to the GenSAA Data Server, and thereby shared with other GenSAA expert systems.

3.6 Benefits of GenSAA

The following benefits are expected to be realized by using GenSAA to build spacecraft monitoring expert systems for future NASA missions:

- Assists the FOAs with data monitoring— FOAs monitor real time data looking for combinations of telemetry parameter values, trends, and other indications that may signify a problem with the satellite or its instruments. The expert systems created with GenSAA will assist the FOAs with the tedious task of data monitoring and allow them to focus on other, higher-level responsibilities during real-time contacts with the satellite. This, in turn, will likely result in more efficient and effective operations.

- Reduces development time and effort; allows quick and accurate response to necessary modifications— The behavior of an orbiting satellite is quite dynamic and occasionally different than anticipated. To quickly create or modify expert systems that can effectively monitor satellites, tools are needed that allow analysts to formulate rule bases easily without the intervention or delay of knowledge engineers and programmers. Several benefits are expected by eliminating these traditional developers. Analysts will be able to create rules quickly in response to unforeseen changes in spacecraft behavior or operational procedures. Also, knowledge translation errors will be reduced or, at least, more easily corrected. Knowledge translation errors are errors which are inadvertently introduced during the process of translating a piece of expert knowledge into rule form.

- Serves as a training tool— In addition to assisting the FOAs with real-time spacecraft operations, GenSAA will be useful as a training tool in two ways. First, by utilizing the playback utilities, analysts will be able to replay a previous spacecraft communications event. Thus, a student analyst can observe how the expert system handles a specific problem scenario. Exercises like this will provide a realistic, hands-on environment for training FOAs in a safe, off-line mode. Second,

experience from previous expert system projects indicates that the development of rules used in an expert system is a beneficial mental training exercise for the FOA. When FOAs create rules themselves, they must consider alternatives more closely and may therefore develop a deeper understanding of the problem domain. This approach may enable more effective fault isolation methods to be identified.

- Protects against loss of expertise— Another benefit of automating fault-isolation tasks with rule-based systems is that the resulting rule base serves as accurate documentation of the fault-isolation method. The rule base can be studied by student analysts to learn about fault-isolation techniques. Even more importantly, mission operations can be better protected against the effects of personnel turnovers. POCC expert systems that capture fault-isolation knowledge preserve expertise from mission to mission and mitigate the impact of the loss of experienced FOAs.

4. Integrating GenSAA Into Other Environments

Even without modifications, GenSAA will readily support the development of highly graphical rule based expert systems. However, in order to receive the "programming-free" benefit that this toolset provides, two steps must be taken: 1) the domain data descriptions must be formatted to allow the Data Manager to display it and thereby facilitate the drag-and-drop interoperability with the other WorkBench tools, and 2) the Data Interface Subsystem of the GenSAA Runtime Framework must be configured to manage the stream of the data selected with the Data Manager. There are basically two approaches for adapting the Runtime Framework for a new environment:

- Modify the GenSAA Data Interface – In this approach, the Data Interface Subsystem of the GenSAA Runtime Framework is modified to accommodate the existing interface of the data source. The advantage to this approach is that the existing data source remains unchanged. However, the disadvantage is that the new user must modify unfamiliar code (GenSAA) and re-implement these modifications for any subsequent GenSAA releases.

- Create a custom Data Server— Perhaps a better approach for integrating GenSAA with the new data source is to create an intermediary process that functions as a TPOCC Data Server from the perspective of the GenSAA Data Interface. This process would receive all data requests from GenSAA and forward all data from the data source utilizing the standard TPOCC interface used by GenSAA. Several advantages would result: the group performing the integration does not have to modify foreign (GenSAA) code, updates to the GenSAA tool will not require re-implementation of the customized portions, and conformance to the original GenSAA Data Interface is maintained. The primary disadvantage is the performance penalty that may result from the extra processing in the intermediary process.

Although the modifications necessary to adapt GenSAA to a new environment may seem to require considerable effort, our experience indicates that it may well be worth the investment. If multiple expert systems are to be developed in a given monitoring environment, the effort required to adapt the GenSAA data interface to this

environment will likely be less than the effort required to integrate each expert system individually with its corresponding data source. The object-oriented design techniques used in GenSAA will simplify modifications and isolate them to specific objects in the GenSAA software.

5. Conclusion

Detecting satellite anomalies is a challenging task that is becoming more difficult as spacecraft become more complex, the number of sensor points multiplies, and data rates increase. As demonstrated by the CLEAR System, fault-isolation expert systems can help human analysts monitor the flood of data. Expert systems can accurately monitor hundreds of real-time telemetry parameters and isolate discrepancies and anomalies the instant they can be detected. They can alert the analysts while providing advice on how to correct problems swiftly and effectively. Unfortunately, development of these systems is often time consuming and costly; moreover, they usually cannot be reused for other missions.

Consequently, GenSAA is being developed for use by the FOAs who work in satellite control centers. GenSAA is designed to enable fault-isolation expert systems to be developed quickly and easily, and without the delay or costs of knowledge engineers and programmers. By facilitating the reuse of expert system elements from mission to mission, GenSAA will reduce development costs, preserve expertise between missions and during periods of personnel turnover, and provide more effective spacecraft monitoring capabilities on future missions. In the commercial sector, similar benefits can be realized with expert systems. Although GenSAA was originally developed to assist with spacecraft monitoring, it naturally supports the rapid development and deployment of graphical intelligent monitoring systems in a wide range industrial applications.

6. References

1. Bielawski, L., & Lewand, R. (1991). *Intelligent Systems Design: Integrating Expert Systems, Hypermedia, and Database Technologies*. New York, New York: John Wiley & Sons.
2. Hughes, P.M. (1989, October). Integrating Expert Systems into an Operational Environment. AIAA Computers in Aerospace VII Conference, Monterey, California.
3. Hughes, P.M., & Luczak, E. (1991, October). GenSAA: Advancing Spacecraft Monitoring with Expert Systems. The American Institute of Aeronautics and Astronautics Computers in Aerospace VIII Conference, Baltimore, Maryland.
4. Hughes, P.M., & Luczak, E. (1991, May). The Generic Spacecraft Analyst Assistant (GenSAA): A Tool for Automating Spacecraft Monitoring with Expert Systems. 1991 Goddard Conference on Space Applications of Artificial Intelligence, Greenbelt, Maryland.
5. Luczak, E.C., Gopalakrishnan, K., & Zillig, D.J. (1989, May). REDEX: The Ranging Equipment Diagnostic Expert System. 1989 Goddard Conference on Space Applications of Artificial Intelligence, Greenbelt, Maryland.
6. Perkins, D., & Truskowski, W. (1990, September). Launching AI in NASA Ground Systems. AIAA/NASA Second International Symposium on Space Information Systems, Pasadena, California.

THE EVOLVING TREND IN SPACECRAFT HEALTH ANALYSIS

Russell L. Kirkpatrick

Jet Propulsion Laboratory
Pasadena, California

N 9 4 - 2 3 9 4 6

ABSTRACT

The Space Flight Operations Center inaugurated the concept of a central data repository for spacecraft data and the distribution of computing power to the end users for that data's analysis at the Jet Propulsion Laboratory. The Advanced Multimission Operations System is continuing the evolution of this concept as new technologies emerge. Constant improvements in data management tools, data visualization, and hardware lead to ever expanding ideas for improving the analysis of spacecraft health in an era of budget constrained mission operations systems. The foundation of this evolution, its history, and its current plans will be discussed.

Key Words: Operations, telemetry, workstations, Unix, vizualization

1. INTRODUCTION

While the goal of space operations is the collection and analysis of scientific data, there is a vast infrastructure required to support that collection. An important part of that infrastructure is the analysis of spacecraft health. Immense volumes of data are tracked by a budget constrained team. An analogous situation is the classic comedy sketch where Hubby goes to get his bowling ball out of the hall closet on bowling night. He opens the door and an avalanche falls on him. The sketch ends by him triumphantly holding up his bowling ball amidst the rubble. Spacecraft analysts have faced their own avalanche by asking for data from their telemetry system. A request for data usually means running a magnetic tape through the telemetry system and handing the results to the requestor who dives in and hopes to emerge triumphantly with the few measurements desired in the first place.

2. HISTORY

The Jet Propulsion Laboratory (JPL) has had several generations of data systems in its support of spaceflight. In the late '60s, IBM 7044s and IBM 7094s comprised JPL's spaceflight data system. About twenty years ago there was a new generation of data system introduced that started a roller coaster of capability for users of these systems.

2.1 Twenty Years Ago

Twenty years ago a new generation of data system was being introduced into JPL's repertoire for spaceflight support. It was comprised of a block redundant IBM 360/75 running a batch orientated operating system modified by the Johnson Spaceflight Center to support real time processing.

While the real time aspects of this data system provided a means of changing which Deep Space Station the data was processed from, there was little to be done to affect the direct processing of the data. Also, while there was a set of displays to choose from, they were hardcoded and required a software delivery for any changes to be incorporated. The only control the users had was which displays were routed to which printers or Digital Television (DTV) Channels (Fig. 1).

There was little in the way of disk storage and the data was written off to magnetic tape to be made into products for the science community. The archive for the engineering community were the printouts and hardcopies of DTV screens that were collected and stored. (Ref. 1) Data reduction of these printouts was comprised of an engineer going through the printouts and either recording or plotting desired values in a journal.

As time went on, the Graphics Display System (GDS) was envisioned and implemented. The GDS allowed individual users to create their own displays with a simple language. They could not control what data flowed through their displays but they could tailor their displays for their own needs. A macro feature was also incorporated into the GDS, called super formats, in which once a set of templates were defined for the display of data, all the user had to do was specify the data and template and not have to get involved with the details.

2.2 Mini-Computers - The Next Generation

In the mid-70s, the arrival of the mini-computer heralded a change to spaceflight support at JPL. It was believed they were so inexpensive that multi-mission systems did not have to be developed anymore, every project could have their own dedicated sets of mini-computers.

It was decided that the Command Subsystem would be moved to mini-computers. About the same time, Voyager decided that they would use their telemetry system that was supporting spacecraft system test for flight support. The result of this move caused the loss of users being able to create their own displays in realtime. This was due too the fact that displays were already hardcoded in the Telemetry System and it was decided to have hardcoded displays in the Command System to reduce cost.

While realtime spacecraft support was regressing, non-realtime was gaining. As flight support was moving off of the IBM 360/75s, they started to be used to make the archive data products. This was the beginning of what has become the Data Records System (DRS). A capability of DRS was to allow a request for a set of data and have that data either printed in a tabular form or plotted. The drawbacks of this system were that it was seven days before the data was available in the DRS, the data usually expired less than thirty days after its receipt on the ground, the request had to be submitted to an operator, and it could take several days to process the request.

2.3 Data Staging

In the early '80s, it was perceived that if data could be kept on disk until processing of it was complete, operations costs could be reduced. Merging files on disk is less labor intensive than hanging magnetic tapes and merging data from magnetic tapes to magnetic tape. Data on disk could be merged from multiple files or data merged into an indexed sequential access method file.

The Multi-Mission Telemetry Transportation Demonstration (MTTD) was a prototyping attempt to design a multi-mission telemetry system that would provide enough disk space to allow data to be kept on disk until the final data products could be generated. Unfortunately, this effort never came to fruition.

After the MTTD had been canceled, the three spacecraft mission Active Magnetosphere Particle Tracer Explorer was approved. For the United States spacecraft it was decided that there would be enough disk space to keep data online until the Master Data Record (MDR) could be generated. The result was that staffing for the creation of MDRs was held to one person. This was in contrast to the estimate of three people during the initial cost estimates for the project. The importance of data staging was realized as a major cost savings when the operational aspects were included in the cost of a project.

For launch, Voyager had used a Univac 1530 to do frame synchronization and error correction. During one of its encounters, the spacecraft data format was changed and could not be accommodated by the Univac 1530 any longer. Also about this time, JPL was to provide a telemetry front end for Ulysses. It was decided that a multi-mission telemetry system would be implemented to do both jobs. The resultant system was the Data Staging and Capture System (DACS). While the DACS was not completely multi-mission, it has been estimated that about 70% of it is common between the two projects. The Voyager DACS provided data to their telemetry system and the Voyager DRS. The Ulysses DACS provides data to their Ulysses

Mission Control System (UMCS) and the Ulysses DRS.

As the name implies, the DACS provided disk for data staging. What the DACS also provides is a data request mechanism. This allows a Voyager Telemetry Operator, a UMCS analyst, or a DRS analyst to request data from one of seven files on disk. As an added value, since data was being loaded in real time, the requesting systems could ask for the file being loaded with real time data and ask for it to be given to them forever. While the controllers of the requesting systems could make data requests of the DACS, the end user, who is the one that really wants to look at the data, still has to make a request to the controllers.

3. SPACEFLIGHT OPERATIONS CENTER

While generations of telemetry systems marched by, technology marched forward. The concept of distributed processing has been used in many parts of the overall ground data system at JPL. These have been in very rigid configurations using point-to-point communication. What had not been done is to distribute that processing to the end user. Not until multi-tasking workstations that could communicate promiscuously with other workstations via a Local Area Network (LAN) were available that the end user could have their own processing to control.

The concept of distributed processing, termed the Space Flight Operations Center (SFOC), communicating over a LAN with data staged and queried by members of the Spacecraft Team from a relational database was developed (Ref. 1,2). Several efforts sprung up in parallel. A Prototype Laboratory was set up to evaluate these new technologies in various combinations. System Engineering developed a System's Concept that utilized the technology that the Prototype Laboratory was to prove.

3.1 The Prototype Laboratory

With the need for new technology to be proven to answer the requirements of a telemetry system, the idea that the concept should be prototyped was developed. Rather than

prototyping a specific concept, it was found that the Prototype Laboratory was prototyping prototyping. There was no specific answer that the Prototype Laboratory was looking for because there were so many options to be conceived and tried. However, there were three areas of concept that the Prototype Laboratory proved that were vital to the creation of SFOC.

One of the early problems the Prototype Laboratory help solve was to find a relational database management system that could handle the data loading rates required and support binary data. Database management systems were primarily updated occasionally and the data queried many times. The concept in SFOC was that the engineering data was constantly being load along with many queries. Also, database management systems handled integers and floating point data but not binary data. The solution found by the Prototype Laboratory was Sybase.

At this time many standards for LANs were evolving. Another concept proved by the Prototype Laboratory was Ethernet running TCP/IP.

Finally, the Prototype Laboratory demonstrated that a workstation running Unix could meet the requirements of SFOC, allow the maximum number of vendors for competition, and not be faced with a solution that leads us to a dead end the future.

3.2 Early SFOC

Magellan was the first project supported by SFOC. It incorporated distributed processing based on workstations running Unix communicating a by LAN based on Ethernet and TCP/IP. The end users, i.e., the Spacecraft Team, could view real time engineering data on their workstations using Data Monitor and Display (DMD) displays they had created (Fig. 2,3).

Additionally, thirty days of engineering data was kept online in a Sybase database management system that they could query. Queries of data eventually fell into two categories (Ref. 3). The first and most

frequent was the standard queries needed for morning report. These queries were incorporated into scripts that would run at night and have the reports produced automatically by the time the Spacecraft Team arrived in the morning. These scripts were eventually made to create a new script for the next nights query and schedule it to fire off. The other type of query, an ad hoc query, was when a problem or other interesting feature was found in the data and a query to look specifically at that data was created and submitted.

3.3 Current SFOC

Mars Observer and Voyager are currently supported by SFOC. Ulysses and Galileo will be supported by SFOC in the near future. These newer projects to SFOC use workstations based on the SPARC microprocessor. Magellan's workstations are based on the Motorola microprocessor and it did not seem prudent to port them to the new architecture. Thus the current SFOC is a second generation SFOC. Although no major changes in the original concept have occurred, a few lessons have learned and have been applied.

A lesson learned from the Magellan experience is that only the most computer literate of the end users were comfortable with ad hoc queries. This second generation SFOC incorporated a new subsystem, the Telemetry Delivery Subsystem (TDS). TDS provides a single interface for acquiring data to the end user whether the data is resident in the database, stored in the TDS cache, or being broadcast over the LAN in realtime.

Another lesson learned from Magellan was that the extra cost of color workstations could be justified for the end users. Instead of using reverse field in displays for flagging alarms (Fig. 1) red alarms could be displayed in red and yellow alarms displayed in yellow. Also windows could be color coded to enhance grouping of like data.

4. FUTURE

Major development of SFOC has ceased and

has entered the sustaining mode. Knowing that there are still capabilities to add and enhancements to be made, SFOC has been renamed the Advanced Multimission Operations System (AMMOS) to indicate the need for these new initiatives to keep the system advanced.

4.1 New Initiatives

One way to use the DMD is to produce a file of processed data. Additional information is being added to this data to make it easier for end users to write their own programs in the analysis of spacecraft health with this data.

One data type that is processed by AMMOS is termed status. The values that a status data type can take is the state of something, e.g., for a tape recorder off, on, record, etc. can be status values. Currently status values can only be displayed in tabular or text form. A Change Request has been submitted to be able to plot these values versus time. A user could then see, over a period of time, the states that some device or measurement has been in (Fig. 4).

A new initiative that has been proposed is for the DMD to accept the input of two data sources simultaneously and through some algorithm keep them synchronized with regards to time. In this way an analyst could forecast the behavior of a given channel and have the forecasted and real values displayed together to see if the spacecraft is following the predicted trend.

Now that AMMOS has been around and used for awhile, the various operational teams and users are being surveyed to see how the system is used and how they would like to use it. Out of this effort new initiatives will be developed and funding will be request requested for their implementation.

4.2 Cassini is Next

The next approved flight project that will adapt AMMOS for its ground data system is Cassini which will visit Saturn.

Cassini is planning for their analysts to have a display of the spacecraft and animate

it as things are happening. The analyst can rotate the image so that all aspects of the spacecraft can be viewed. Color can be used to indicate where measurements in alarm actually reside on the spacecraft or indicate temperature.

The ability to acquire a set of interesting data is of little use if one is merely left with rows of numbers to peer at. Comprehension has been improved through the use of tailored displays for viewing individual measurements, the use of emerging visualization techniques such as spectrographs, and the use of an analyst's own software. The increased capability of visualizing data is, in large part, again due to the distribution of computing power to the end users. Innovative visualization of data continues to be of importance in the ongoing development of this evolving mission operations system and new technology is being introduced as it becomes available.

5. REFERENCES

1. Bahrami, K. A., et al. 1988. Automated Workstation for the Operations of Spacecraft Engineering Subsystems. In *Information Systems Prototyping and Evaluation Quarterly Report No. 3*, Flight Projects Office Information Systems Testbed, Jet Propulsion Laboratory, Pasadena, California.
2. Burleigh, Scott. 1992. Architecture Study for the Multimission Spacecraft Analysis. In *Information Systems Prototyping and Evaluation Quarterly Report April 1992*, Flight Projects Office Information Systems Testbed, Jet Propulsion Laboratory, Pasadena, California.
3. Bucher, A. W., et al. 1992. Using SFOC to Fly the Magellan Venus Mapping Mission. In *Proceedings of SpaceOps 1992*.

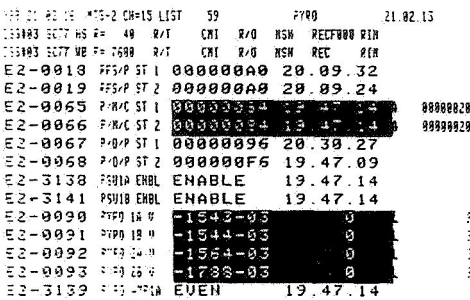


Fig. 1 - DTV Display

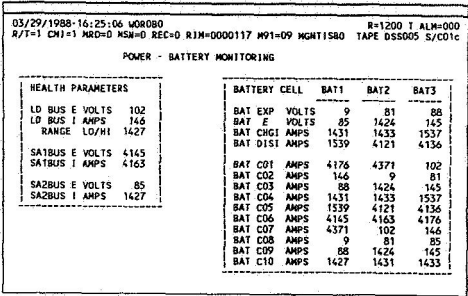


Fig. 2 - DMD Fixed Display

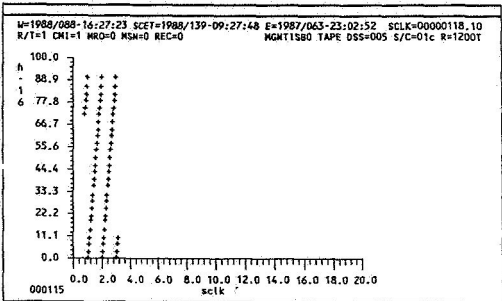


Fig. 3 - DMD Channel vs. Time Plot

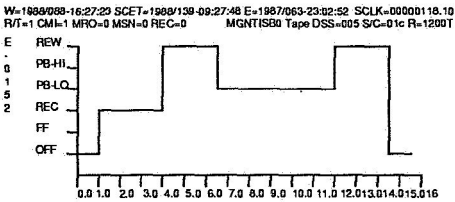


Fig. 4 - Status vs. Time Plot

ORBITAL SIGNATURE ANALYZER (OSA): A SPACECRAFT HEALTH/SAFETY MONITORING AND ANALYSIS TOOL

Steven Weaver, Charles DeGeorges, and Joy Bush
Computer Sciences Corporation
1100 West Street, Laurel, MD 20707, USA

N94-23947

Robert Shendock
TRW, Incorporated
One Space Park, Redondo Beach, CA 90278, USA

Daniel Mandl
NASA/Goddard Space Flight Center
Code 511.2
Greenbelt, MD 20771, USA

ABSTRACT

Fixed or static limit sensing is employed in control centers to ensure that spacecraft parameters remain within a nominal range. However, many critical parameters, such as power system telemetry, are time-varying and, as such, their "nominal" range is necessarily time-varying as well.

Predicted data, manual limits checking, and widened limit-checking ranges are often employed in an attempt to monitor these parameters without generating excessive limits violations. Generating predicted data and manual limits checking are both resource intensive, while broadening limit ranges for time-varying parameters is clearly inadequate to detect all but catastrophic problems.

OSA provides a low-cost solution by using analytically selected data as a reference upon which to base its limits. These limits are always defined relative to the time-varying reference data, rather than as fixed upper and lower limits. In effect, OSA provides individual limits tailored to each value throughout all the data. A side benefit of using relative limits is that they automatically adjust to new reference data. In addition, OSA provides a wealth of analytical by-products in its execution.

Key Words: Orbital signature, limit sensing, telemetry, spacecraft

1. INTRODUCTION

The responsibility of the control center is to maintain the health and safety of the spacecraft. Controllers utilizing traditional tools routinely perform a real-time rudimentary state of health assessment. This assessment is only as good as the available tools, analytical quality of the databases they access and the controllers that use them. Early detection and response to potential problems increases the probability of meeting mission objectives.

Traditional tools are limited and resource intensive. Only limit sensing provides a broad continuous assessment of spacecraft condition on the ground. While accurate assessment of bi-level states and slightly varying analog parameters can be achieved, tight limit definitions are required. Frequent database updates supported by intensive data analysis is needed to maintain the limits. Assuming this can be routinely supported, accurate assessment of widely varying parameters is seldom achieved. An example of a parameter in this class would be solar array current for a low earth orbiter. Despite the fact that the parameter is well behaved and shows a distinctive orbital character, steps are taken to avoid frequent erroneous limit violations. This is usually done by setting very wide general limits, then applying techniques (i.e., conditional limits, etc.) to better evaluate the slightly varying segments of the data. This process often results in some invalid limit violations and loose evaluation of the parameter over much of the orbit.

Despite best efforts, the probability of early problem detection is not what it should be. To compensate,

additional engineering analysis is performed to supplement the real-time evaluation. The analysis is done to detect potential problems and assess system performance. Short, medium and long term trends are evaluated against a reference or the expected. Unfortunately, lag times and resource requirements are high in generating the analysis and transferring analytical products to the control center.

2. OSA CAPABILITIES

OSA was developed to solve these problems using an approach as old as control centers themselves. Traditional evaluation often involved a technique called "holding two overlapping plots up to the light". This produced a qualitative analysis not directly storable on magnetic media. OSA takes a more integrated and exact approach. The system performs a quantitative analysis of current versus reference data in real-time or batch modes. The reference data is analytically determined and may come from any source (i.e., simulator, s/c tape dump, etc.). OSA graphically displays reference data, current data and user defined variable limits on a single plot. More importantly, differences between the current and reference data are calculated and plotted with user defined difference limits. Limit sensing is performed on the difference data, violations are reported to the controller in real-time and violation statistics are maintained. The process significantly increases the control center's ability to accurately assess widely varying telemetry. This is accomplished by sensing against an analytically determined varying limit definition.

OSA is used to set new limit definitions upon user request. The reference data is graphically displayed and the user is allowed to segment the data. High and low limits are defined for each segment relative to the reference data values. Two types of limits can be defined. The most frequently selected is a percentage of each individual reference value. The other option is to apply a fixed offset to each reference point. Limit values are usually reviewed upon update of the reference image.

OSA allows the user full management of the reference image. Typically, candidate reference data is analyzed by OSA and optionally used to update the reference image upon engineer approval. Limit definitions can be modified. Limit values automatically adjust to the new reference data. The reference image and limit definitions directly reflect the current engineering requirements. Thus,

detailed analysis can efficiently be transferred to the Control Center for real-time use in a painless manner.

3. OSA BENEFITS

One indicator of a tool's worth is the amount of unexpected benefit that falls out from its use. While detecting several problems and identifying other curiosities, OSA output supports most GRO documentation concerning the power subsystem where detailed analysis of orbital behavior is required. The OSA archival function has been used extensively to compare selected sets of data separated in time but collected under similar spacecraft conditions. In addition OSA contributes to medium and long term statistical data calculation.

The traditional statistics are calculated for the current data and include minimum, maximum, mean, standard deviation and variance. Uniquely, statistics are calculated for the difference data and include average differences, number of samples out of tolerance and the maximum number of consecutive samples out of tolerance. Statistical values for all parameters under evaluation are displayed on Statistical Summary Pages. These pages are routinely maintained during a contact in the real-time environment. Limit violations are indicated on these pages via a color coding scheme. The graphical displays are available and contain the statistics for the plotted parameter. All statistical data is available in standard format via magnetic media for subsequent analysis.

OSA has proven itself by the early detection and detailed analysis of the GRO (Gamma Ray Observatory) MPS-1 battery problems. While two missions were experiencing similar problems, GRO was able to clearly identify the problem early on and define even slight character changes in battery state of health indicators. To this date, GRO still provides the most detailed data analysis of early battery degradation character. This, of course, is important to the future. OSA's capabilities are required for the earliest detection of problem symptoms.

The control center's experience with OSA has been very rewarding. This is due to the soundness of the concept, the quality of the products and the professionalism of the developers.

4. OSA IMPLEMENTATION

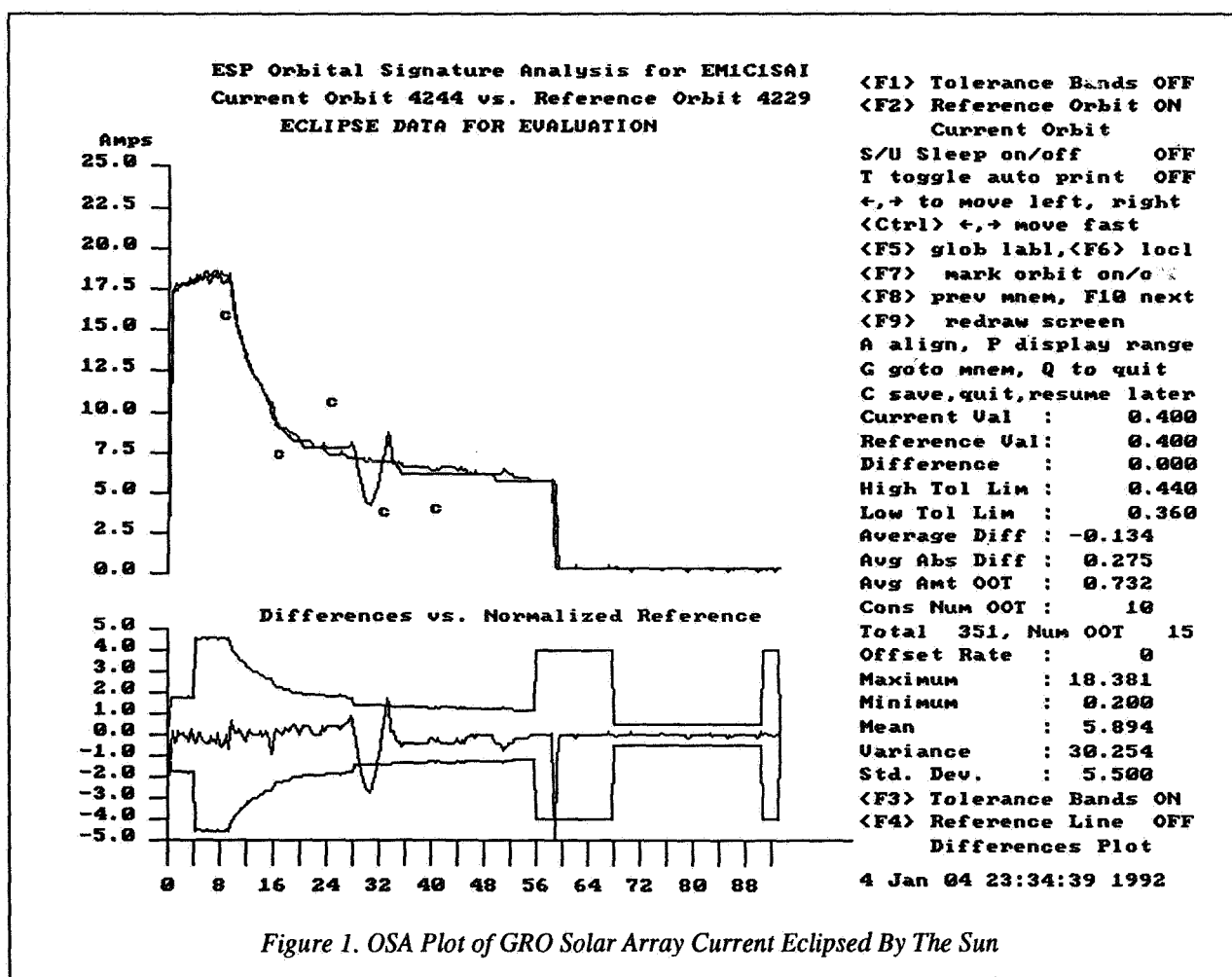
At start up, OSA menu options allow the FOT to provide default settings for manual or automatic ("sleep mode") analysis and print operations. During operation, OSA settings, display ranges, manual or sleep mode operation, and navigation among mnemonics are controlled through function keys.

The PC version of OSA uses C along with an assembly language-based commercial graphics package to immediately display or modify graphical elements such as orbital signatures, associated tolerances, moveable timelines, or tolerance band boundaries, even on a 286-based PC. Display ranges for parameters may be modified, and the current data may be manually aligned with the reference to further refine the automatic alignment performed to eliminate the effects of nodal precession on the current data.

The TPOCC (Transportable Payload Operations Control Center) version of OSA is being developed for Hewlett Packard and Sun workstations in C++, using OSF/MOTIF, and is X-Window based.

5. OSA EVOLUTION AND FUTURE DIRECTIONS

OSA is intended to be a real-time telemetry monitoring tool for time-varying parameters such as power system data for the GRO. OSA was to use neural networks trained on orbital signature examples that indicated developing problems. In this way, OSA could examine either tape recorder playback or real-time pass data and flag impending problems BEFORE they exceeded critical limits. The neural networks were to be trained on simulated or hand-estimated data generated before launch. Thus the origin of the currently used acronym of OSA in the GRO - ESP expert system predictor (as shown on Figure 1).



Five key lessons learned are shown below that describe both the development process as well as the user (and non-user) reaction to OSA.

LESSON 1 - *A model-based system needs a model*

Simulators are wonderful tools, but for many missions and for a variety of reasons, simulators are not able to supply the necessary training data for the neural networks. For the GRO, the spacecraft analyst felt that the predicted power system values, while reasonable, were not sufficiently certain to be used as the basis for neural network training. However, the concept of a reference orbit, as a basis for comparing current data, was kept.

As delivered, OSA uses a previously captured (and reviewed and approved) orbit's data as a reference for comparison to the current. Operationally, the reference for the GRO is updated daily using the daily orbital data after comparison to the previous reference orbit. This works well in practice because day-to-day orbital signature variations are small. Ultimately, OSA will still use neural networks trained on flight data along with auxiliary information to recognize developing problems on-board the spacecraft, thus living up to its name, ESP.

LESSON 2 - *Prototype, demonstrate, and change - continually*

From the developer's perspective, OSA is a success story due in part to excellent customer feedback from the many prototype demonstrations performed, but also due to constant incorporation of changes or new features. No feature, tool, or approach was held sacred except the final objective of the system itself. Changes were so extensive that the completed offline PC-based version of OSA incorporated neither neural networks nor used simulator data as originally planned but, most importantly, it met the user's data monitoring and analysis needs.

LESSON 3 - *For analysts, quantity is more important than quality*

Analysts make critical spacecraft operational decisions based on the data supplied by various tools. They will make use of qualitative assessments if they are given the numbers as well. If only one of the two can be provided, they want the numbers instead of the tool's "expert" assessment of what is happening on-board the spacecraft.

OSA/ESP is trusted because it provides analyst-selected quantitative measures of the spacecraft power system state of health based on analyst-assigned tolerances employing an analyst-approved reference orbit.

LESSON 4 - *The FOT may dislike what the spacecraft analyst loves*

The flight operations team and spacecraft analysts have different needs. The FOT user desires automation of routine activities that include data capture, comparison, analysis, and plotting, and print and archive. FOT interaction is desired only when exceptions are reported that highlight potential problems. To that end, saved menu selections, predicted event file inputs, statistical summary pages and color coded limit sensing are provided by OSA. OSA automatically displays and prints statistics as well as plots.

The spacecraft analyst, on the other hand, may wish to investigate specific problems, such as out-of-tolerance conditions, spotted by the FOT. OSA manual navigation features include moving among mnemonics, displaying a specific mnemonic, save and resume capability, select and set tolerance bands and values, modifying display ranges, tolerance thresholds, manually aligning current and reference orbit signatures, and switching on or off any of the automated functions.

We learned that, if in doubt about which features to provide to the typical spacecraft analyst, at a minimum, do the following:

- Capture and plot everything
- Compute all statistics
- Archive and print everything
- Process previously archived data
- Output data and statistics in "standard" formats, ASCII, and format for "standard" reports as well
- Allow any or all of these features to be toggled on or off
- Allow saving of feature settings

- Allow for batching or grouping of various menu commands to permit hands-off (a.k.a. "sleep mode") operation
- Anything else thought of during the prototyping and testing process

On the other hand, for the FOT, the rules are simple. For a tool fated for control center use ensure that the tool does the following:

- Does not require frequent tending
- Output is reliably correct
- Return exceeds the effort

LESSON 5 - *The OSA concept is too simple to understand.*

We've demonstrated OSA dozens of times both to those with control center experience (both FOT members and spacecraft analysts), and to those with less familiarity with spacecraft operations. Because OSA also happens to plot orbital data and generate statistics, it is often mistaken for a trending and/or plotting package. These and other common reactions are listed below followed by a summary of our typical response.

- Doesn't standard limit checking do this checking already? *No, it essentially just performs catastrophic limit checking for time varying parameters.*
- An OSA out of tolerance condition does not indicate a problem if the value is still within the static limits. *It most certainly does (we cite the GRO battery and the GRO lunar eclipse example).*
- We already have a plotting and/or a trending package.
- We already plot out all data for critical parameters.
- We already generate standard statistics for all parameters.
- We already look at the plots of problem parameters.

Rarely, if ever, are the four items above done in a single package either in real-time or in background. They are by-products of OSA, not the product.

- We don't really need OSA because the only thing that it has that our tools currently do not is this idea of a reference. *You still don't routinely (daily and each pass) align and compare all critical parameters to their respective references and produce a quantitative assessment.*
- Isn't there too much change in the orbital signature daily? *No, while the daily change is a fraction of each time-varying parameter's static limit range, future enhancements will analytically verify changes in spacecraft profile allowing OSA to expertly adjust tolerances.*
- OSA runs on a PC and will soon run on Sun and HP workstations. Could we use it in our control center without many/any changes ? *Quite possibly.*

6. ENHANCEMENTS UNDERWAY

Present enhancements currently under development include an expert system to modify the reference curve and current orbit data according to the operational state of the spacecraft, thereby accounting for the effect of spacecraft activities during curve comparison. Closer alignment of the two curves permits even tighter tolerances to be assigned. The state change rules are user definable. The reference orbits may be hand modified using mouse input. Also, patterns of interest may be identified using the mouse and searched for by neural networks trained on the patterns.

Finally, the system is being integrated into a new NASA/Goddard TPOCC, and will be object oriented and Motif compliant, for use on such missions as SAMPEX and Wind/Polar.

SESSION K

TELEMETRY PROCESSING, MISSION DATA MANAGEMENT, AND DATA ARCHIVING

CO-CHAIRPERSONS

M. M. Ebersole, JPL, California Institute of Technology, USA
G. A. Smith, NASA Goddard Space Flight Center, USA

Summary	765
M. M. Ebersole	
K.1 EOS Ground Data Systems: A Description and Interface Overview	767
G. Smith, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA	
K.2 Telemetry Ground System Development via the Component Approach	773
K. E. Thorn, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA	
K.3 Decentralized Architecture for Data Processing in the In-Orbit Infrastructure Ground Segment	779
P. Poirier, Matra Marconi Space France, Toulouse, France	
K.4 Using the SPICE System to Help Plan and Interpret Space Science Observations	781
C. H. Acton, Jr., JPL, California Institute of Technology, Pasadena, California, USA	
K.5 An Engineering Database Management System for Spacecraft Operations	787
G. Cipollone and M. H. McKay, ESA European Space Operations Centre, Darmstadt, Germany; J. Paris, BIM, Everberg, Belgium	
K.6 The ESIS Query Environment Pilot Project	791
J. J. Fuchs, Computer Resources International A/S, Bregnerødvej, Denmark; A. Ciarlo, ESA European Space Research Institute, Frascati, Italy; S. Benso, Compagnia Italiana Servizi Tecnici (CISIT) S.p.A., Space Division, Rome, Italy	
K.7 Mars Observer Data Production, Transfer, and Archival: The Data Production Assembly Line	803
D. B. Childs, JPL, California Institute of Technology, Pasadena, California, USA	
K.8 The European Space Agency Standard for Space Packet Utilisation	813
J.-F. Kaufeler, ESA European Space Operations Centre, Darmstadt, Germany; A. Parkes, NOVA Space Ltd., Bath, UK; A. Pidgeon, VEGA Space Systems Ltd., Harpenden, UK	
K.9 Multimission Image Processing and Science Data Visualization	819
W. B. Green, JPL, California Institute of Technology, Pasadena, California, USA	

K

K.10 Compression for an Effective Management of Telemetry Data823

J.-P. Arcangeli, Institut de Recherche en Informatique de Toulouse & Greco de Programmation du Centre National de la Recherche Scientifique (CNRS), Toulouse, France; M. Crochemore and J.-E. Pin, Laboratoire d'Informatique Théorique et Programmation & Greco de Programmation du CNRS; J.-N. Hourcastagnou, CNES Toulouse Space Centre, France

K.11 Space Station Freedom Ground Data System: Design and Operations831

R. A. Dunning, Jr., Booz-Allen & Hamilton Inc., Washington, D.C., USA;
F. W. Knops, III, Booz-Allen & Hamilton Inc., Seabrook, Maryland, USA;
S. A. Fishkind and M. P. Pasciuto, NASA, Washington, D.C., USA

K.12 Earth Observation Archive Activities at DRA Farnborough837

M. D. Palmer, Defence Research Agency, Space Sector, Farnborough, Hampshire, UK; J. M. Williams, Serco Space, Southall, Middlesex, UK

SUMMARY — SESSION K

M. M. EBERSOLE
Jet Propulsion Laboratory

Session K provided an interesting set of papers on a variety of subjects. Of the 12 scheduled papers, 10 were given. The papers can be grouped by the following topics:

- Telemetry — four papers, ranging from the discussion of very large, complex telemetry systems down to telemetry system components.
- Mission data management: four papers.
- Data archival: two papers.

Several common themes emerged. All the papers showed sensitivity to the current cost climate. Most papers were geared toward providing more efficient operations development or reduced operations staffing to reduce cost.

Many papers recognized the need to treat data as an important resource. Data should be easily accessible, understandable, and usable. Other themes were also evident and are described below.

Significant technology trends were evident. For example, rapid increases in processing power and communications services continued and furthered the appeal of distributed systems; gains in the density of data storage produced an increase in systems storage capacity and increase in data transfer rates; and very large-scale integrated (VLSI) circuitry for high-performance telemetry data-handling functions permitted reductions in the cost and size of data acquisition and processing systems. Other trends were shown: workstations continued to proliferate and grow in power. Trends noted were:

- *The expanding role of telecommunications.* Increasingly, individual scientists, computers, and archives are linked by high-speed electronic and fiber-optics communications networks. These networks allow greater numbers of operations and processes to be performed remotely.
- *Technological advances in high-performance subsystems based on VLSI components.* Advances in gate arrays and microprocessor technology are critical to meet the high data rate requirements over the next decade and to reduce systems implementation costs.
- *Advances in artificial intelligence, including expert systems and neural networks.* Adaptation of this technology to information systems and operations has a high potential for automating the more routine aspects of data processing and control center operations, for screening large volumes of data, and for capturing the knowledge required to interpret complex information products.
- *Enhancement of institutional service capabilities.* Economies of scale continued to be gained through provision of standard, large-scale communications services to a variety of users. Examples include the TDRSS with its successor, Advanced-TDRSS; the DSN and communications backbones such as the NASA Science Internet (NSI); and the emerging National Research and Education Network.

The industry trend is toward open systems, the focus for which includes:

- *Portability* — the ability to use application software on computers from various vendors. The advantage to us, of course, is that we will be able to select from a wide range of vendors and make choices based on the best price performance.
- *Interoperability* — the ability to have computers from different vendors work together. This is the “glue” that enables the distributed approach.
- *Scalability* — the capability to use the same software environment on many classes of computers, from PCs to supercomputers.

N94-23948

EOS Ground Data Systems: A Description & Interface Overview

Gene Smith
EOS Data Systems Manager
NASA/GSFC
Mail Code 502.2, Greenbelt, MD 20771
Tel. (301) 286-4285; Fax (301) 286-5269

ABSTRACT

The Earth Observing System (EOS) is planned as a space-based measurement system, earth-science research program, and data and information system (EOSDIS). It will consist of several high data rate spacecraft with multiple earth sensing instruments which provide investigators with a thorough, long-term view of the earth's environment. Up to seven spacecraft may be supported at once, either in operational, checkout, or testing phases; and the average data rate from the EOS satellites in orbit at any one time is expected to be from 18 to 60 Mbps.

Providing the data processing and flight operations support for EOS will be the EOSDIS Core System (ECS). The ECS will command and control the spacecraft; process and store the EOS data; provide access to the data for years; and support researchers. The data processing aspects of the ECS consist of a collection of Distributed Active Archive Centers (DAACs) which perform the product generation, data archive and distribution, and information management services. Flight operations aspects will be provided by the EOS Operations Center, by instrument control centers, and by widely distributed instrument support terminals. The communications and system management aspects will be provided by the EOSDIS Science Network and the System Management Center. In addition to the EOS satellite data, other data sets from earlier earth science missions are also to be added to designated DAACs.

Other ground data systems which will provide support to EOS for acquiring, transporting, processing, and distributing the transformed spacecraft data are currently being defined or are being upgraded for the EOS era. These systems include the Space Network consisting of the Tracking and Data Relay Satellite System (TDRSS), the TDRSS Ground Terminals, and the Network Control Center as well as the Flight Dynamics Facility, the EOS Data and Operations System, and EOS Communications.

This paper briefly describes data handling by the ECS, the support data systems, their interfaces, and their roles.

Key Words: Earth science, data systems, data processing, flight operations, communications, information systems

1. BACKGROUND

NASA's future role in space has been focused in the 1990s on planetary and astronomical investigations—Mission from Planet Earth—and on developing a detailed understanding of our own planet—Mission to Planet Earth (Fig. 1). This is NASA's response to the U.S. Global Change Research Program involving an integrated effort of nine U.S. Government agencies. The Mission to Planet Earth has many components including NASA's Earth Observing System (EOS). The EOS consists of space-based measurement systems, a data and information system, and scientific research programs.

Beginning in the late 1990s, EOS will utilize large, multi-instrument spacecraft to gather data about the earth in order to coordinate and extend investigations of the earth's atmosphere, oceans, ice caps, land, and water resources. Tens of megabits of data will be generated every second, and petabytes (10^{15} bytes) of data will be stored on the ground over years for access by investigators. Scientists will analyze observations from EOS, combine them to achieve interdiscipline synergism, and coordinate their findings with ground-based observations and other satellite measurements to expand the level of understanding of all earth science.

After observations are made by an EOS spacecraft and before the analyses are released, a variety of crucial data transformations will be made to the data to produce new information on earth phenomena. To do this for the volumes of data to be gathered will require an extensive ground support system to command and control the spacecraft and instruments, transport the observation data, process the data into standard products, store the data products, provide access to EOS information for years, and support researchers in their investigations. This ground support system is collectively known as the EOS Data and Information

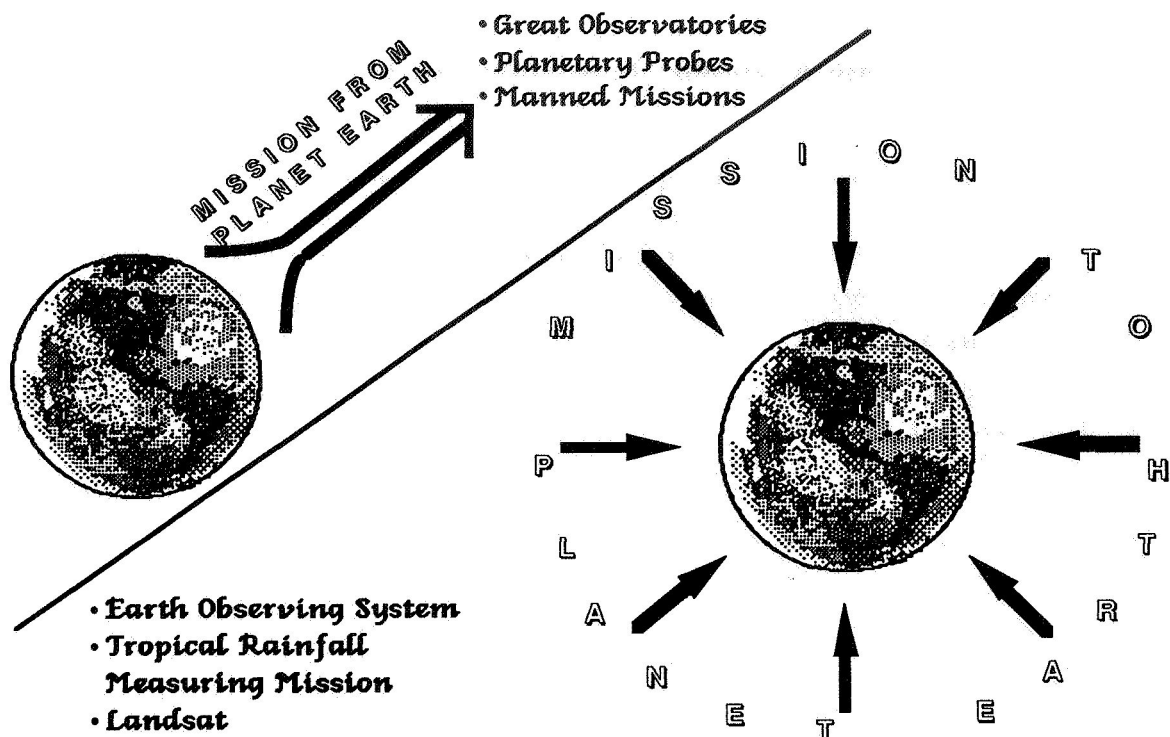


Fig. 1 — NASA Missions

System (EOSDIS) and will consist of a distributed network of facilities and functions. Providing standard EOSDIS services and products to the earth science community will be the EOSDIS Core System (ECS). NASA institutional facilities will also be utilized to provide standard NASA services to the ECS.

The ECS will consist of three segments: Flight operations, science data processing, and communications and system management. These are divided into elements which are to perform related functions distributed to a number of sites at centers of scientific expertise.

2. EOS DATA HANDLING

2.1 The EOS On-Board Data System

The EOS on-board data system collects data packets from the instruments and spacecraft sensors at asynchronous rates. These packets contain radiance measurements, image scan lines, or other sensor-unique observations. The on-board data system merges low-rate packets into virtual data channels by breaking each merged data stream into synchronous frames with a set number of bits, adding framing headers and providing error protection encoding of

each frame. The low-rate instrument virtual channels are further merged with high-rate instrument virtual channels for transmission to the ground or for recording for later transmission.

Uplink data (commands and memory loads) arrive at the spacecraft in coded virtual channels. Then the virtual channel formatting is removed, the data decoded, and the commands are sent to the appropriate instrument or spacecraft subsystem for interpretation.

The average data rate for the EOS A.M. spacecraft is 18 Mbps with first launch scheduled for mid-1998. Subsequent spacecraft in the A.M. series (three spacecraft at five year intervals) and the spacecraft in the P.M. series (three spacecraft at five year intervals beginning in late 2000) are expected to produce equal or greater amounts of data. Additional EOS spacecraft are the Chem, Aero, and Alt series with fewer instruments and lower aggregate data rates. The EOS is planned for 15-20 years of active data collection; and support for access to the EOS data archives could continue indefinitely, depending on user interest and funding. Landsat 7 and subsequent Landsat flights are also planning to utilize the EOSDIS ground data systems.

2.2 NASA Institutional Facilities

Several NASA institutional facilities contribute to getting the data packets created by the instruments to the ECS for extensive processing and to moving the commands and memory loads from the ECS to the spacecraft. These facilities include the Space Network (SN), NASA Communications (Nascom), and the Flight Dynamics Facility (FDF).

The SN consists of the Tracking and Data Relay Satellite System (TDRSS) and a Network Control Center (NCC). The TDRSS acquires the EOS spacecraft data transmissions at one of its synchronous-altitude satellites (TDRS) from an EOS spacecraft and retransmits the data to one of two TDRS Ground Terminals (TGTs) at White Sands, New Mexico. Transmission artifacts are removed at the TGT. In addition, commands are uplinked through the TDRSS to the spacecraft. The NCC schedules and controls the TDRSS links and the data traffic through them. Development of an advanced TDRSS (TDRSS II) is now underway for enhanced operation in the next decade.

The Flight Dynamics Facility (FDF) will provide orbit support for EOS by developing and managing the TDRS On-board Navigation System (TONS). TONS will compute the EOS spacecraft position, velocity, and time parameters on-board the spacecraft from two-way coherent tracking measurements and Doppler extraction between the spacecraft and the TDRSS satellites. In addition, the FDF will provide and maintain attitude sensor calibrations for on-board attitude determination. The FDF will monitor the on-board position and attitude computations, determine any corrections to be made to the on-board algorithms, and provide spacecraft attitude and orbit control information to the EOS Operations Center.

Nascom is a worldwide complex of communication circuits and systems which support NASA programs. For EOS, a dedicated communications network (affiliated with Nascom) provides the ground communication circuits for the spacecraft telemetry data and the command data. Nascom supplies voice circuits between systems and provides the circuits for communication by the EOS Operations Center with the NCC and FDF.

2.3 EOSDIS Elements

The EOSDIS will be developed in an incremental and evolutionary manner. In addition to the ECS, it will include an early phase (Version 0), a testing component for Independent Verification and Validation, and a dedicated data capture and processing service—the EOS Data and Operations System

(EDOS). A dedicated communications network for telemetry and command data—EOS Communications (Ecom), interconnection with institutional data transport systems, and science data product algorithm development—Scientific Computing Facilities (SCFs) will also be included in the EOSDIS. The relationship of the ECS to other EOS Ground Systems is shown in Fig. 2.

EOSDIS Version 0 was initiated in 1991, using existing earth science data systems to provide interoperability and improve access to existing earth science data. Version 0 will be developed prior to and outside the ECS contract with development of data sets derived from past and current earth science missions. It will also develop a substantial information management service, provide limited product generation, and develop substantial data archive and distribution capabilities. It will support commonality among cooperating data systems; develop prototype technologies; implement standards, protocols, and guidelines; and provide a unified earth sciences view to its users. Results from these activities will be available in the development of the ECS.

The EDOS will decode the data, flag error conditions, remove virtual channel framing artifacts, and reassemble the packets at White Sands, New Mexico—collocated with the TDRSS Ground Terminals. The packets are transmitted to Fairmont, West Virginia, for further processing. At Fairmont, the EDOS will produce data sets of time-ordered, nonredundant data packets for each instrument; transmit the data sets to the ECS; and archive all data sets for protection. If faster throughput is required for some instrument data, a reduced set of operations may be selected. In addition, commands arrive from the ECS in packet streams. These packets are formatted into virtual channels and passed to the ground station for uplink to an EOS spacecraft.

Ecom will transmit the data sets from White Sands to Fairmont, West Virginia, and from Fairmont to the appropriate ECS processing sites and will transmit command packets from the ECS to White Sands.

The primary functions of EDOS and Ecom are shown in Fig. 3.

3. ECS—GROUND DATA PROCESSING AND FLIGHT OPERATIONS

The ECS is the central feature of the EOSDIS. It will be designed, developed, built, and operated over the next ten years through a NASA contract to a team of aerospace corporations. The three ECS segments—flight operations, science data processing, and

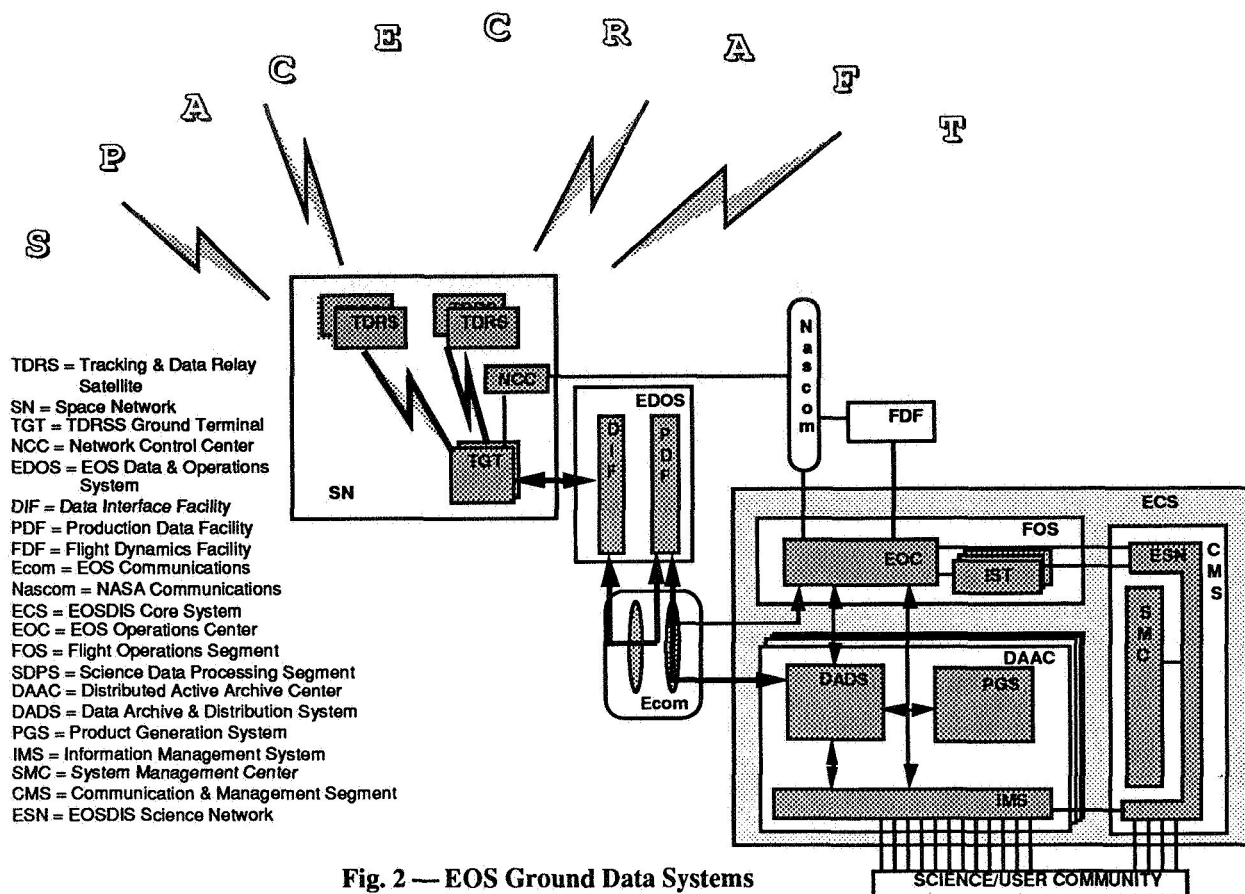


Fig. 2 — EOS Ground Data Systems

communications and system management—are divided into elements which perform related functions distributed to a number of sites at centers of scientific expertise.

3.1 The Flight Operations Segment

The Flight Operations Segment of the ECS will control the spacecraft, provide mission planning and scheduling, and monitor the health and safety of the spacecraft and instruments. The primary element of this segment is the EOS Operations Center, which will be responsible for the health and safety of each spacecraft; support planning and scheduling of platform resources; coordinate instrument observation requests; and validate commands to instruments. It will also serve as the U.S. Instrument Control Center (ICC) to plan and schedule each instrument's operations and to generate appropriate commands. It will also be responsible for instrument health and safety. For the International Partner (IP) instruments on U.S. spacecraft, additional ICCs may be located at the IP sites as negotiated. These would be linked to the EOC through Ecom. The Instrument Support

Terminal element is distributed to key investigator facilities to perform instrument calibration; monitor instrument performance, health, and safety; resolve performance anomalies; and contribute to instrument planning.

3.2 The Science Data Processing Segment

The Science Data Processing Segment of the ECS processes the incoming science packet data sets into information in which spectral, temporal, and spatial observations are transformed and combined to obtain new parameters, images, or maps. There have been eight sites selected to perform this science data processing and other data services. They are Distributed Active Archive Centers (DAACs). Each DAAC has three elements. A Product Generation System (PGS) produces standard data products from approved algorithms. A Data Archive and Distribution System (DADS) stores the data products, distributes them to investigators, and makes the data available for decades for subsequent research. An Information Management System (IMS) monitors additions to and status of the data products in

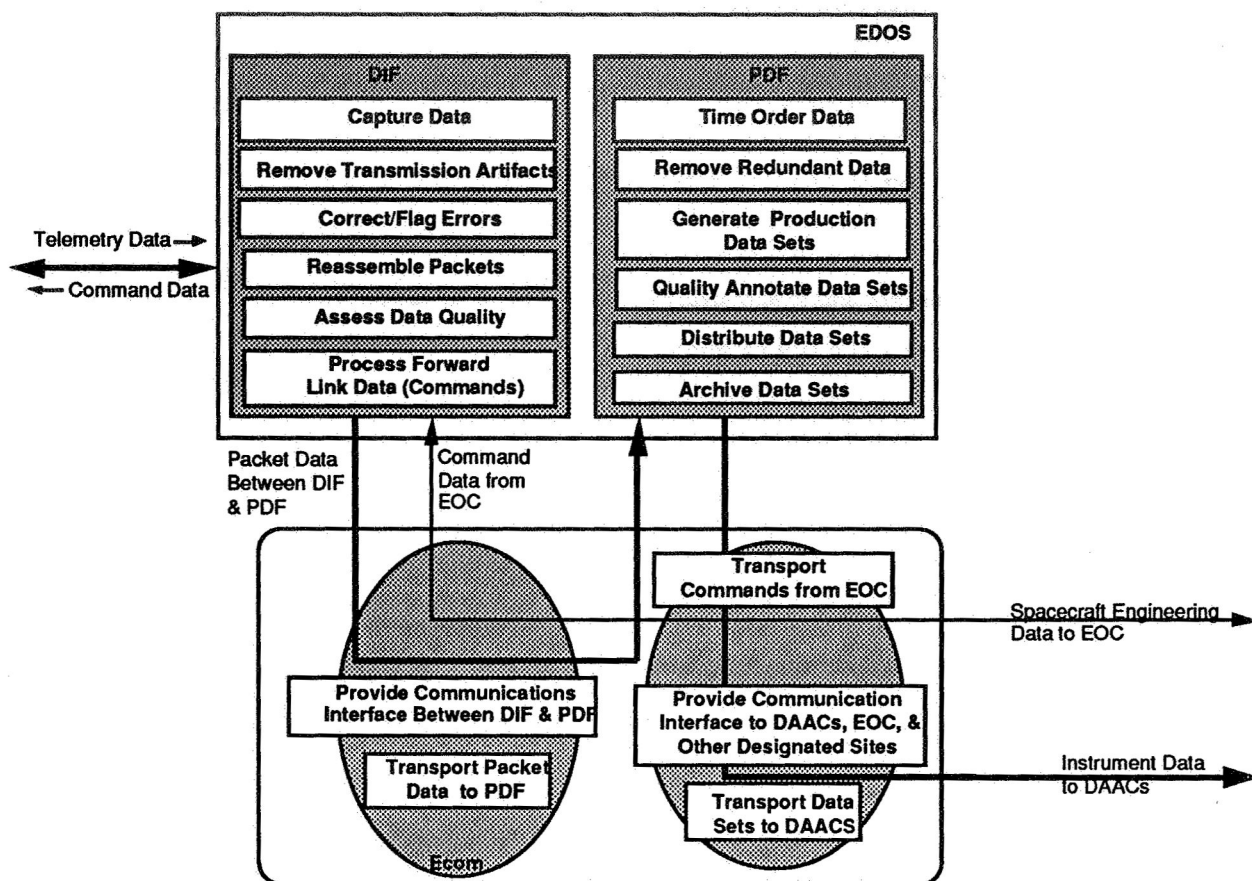


Fig.3—EOS Data & Operations System and EOS Communications Functions

the archive, makes the data product and data inventory information available for access by local and remote users, and provides these services for all DAAC archives through access to any DAAC. Technical assistance and automated help support will be provided to assist users requesting information. The highlights of Science Data Processing functions are shown in Fig. 4.

3.3 The Communications and System Management Segment

The Communications and System Management Segment has two elements. The EOSDIS Science Network distributes data among the DAACs and to users over NASA-provided circuits while the System Management Center provides on-line system status information of all ECS elements; configuration and performance management; high-level ECS scheduling, monitoring, and accounting; security management; and user authorization and billing. The

Science Network also provides network management and user assistance. Related to the EOSDIS Science Network but outside the ECS and serving to support ECS services is the NASA Science Internet to interconnect users with the DAACs.

3.4 ECS Goals

These elements serve to provide the functionality required of the ECS to support the EOS instrument investigators, earth scientists, and other researchers. Included in the ECS archives will be the EOS spacecraft data and other earth research satellite data sets to provide the investigators with a rich mixture of information. Through these elements, the EOSDIS and the ECS will meet the goals of supporting system evolution; providing elegant access by investigators to massive, growing data stores; facilitating interdisciplinary research, and enabling increased understanding of our earth and its environment.

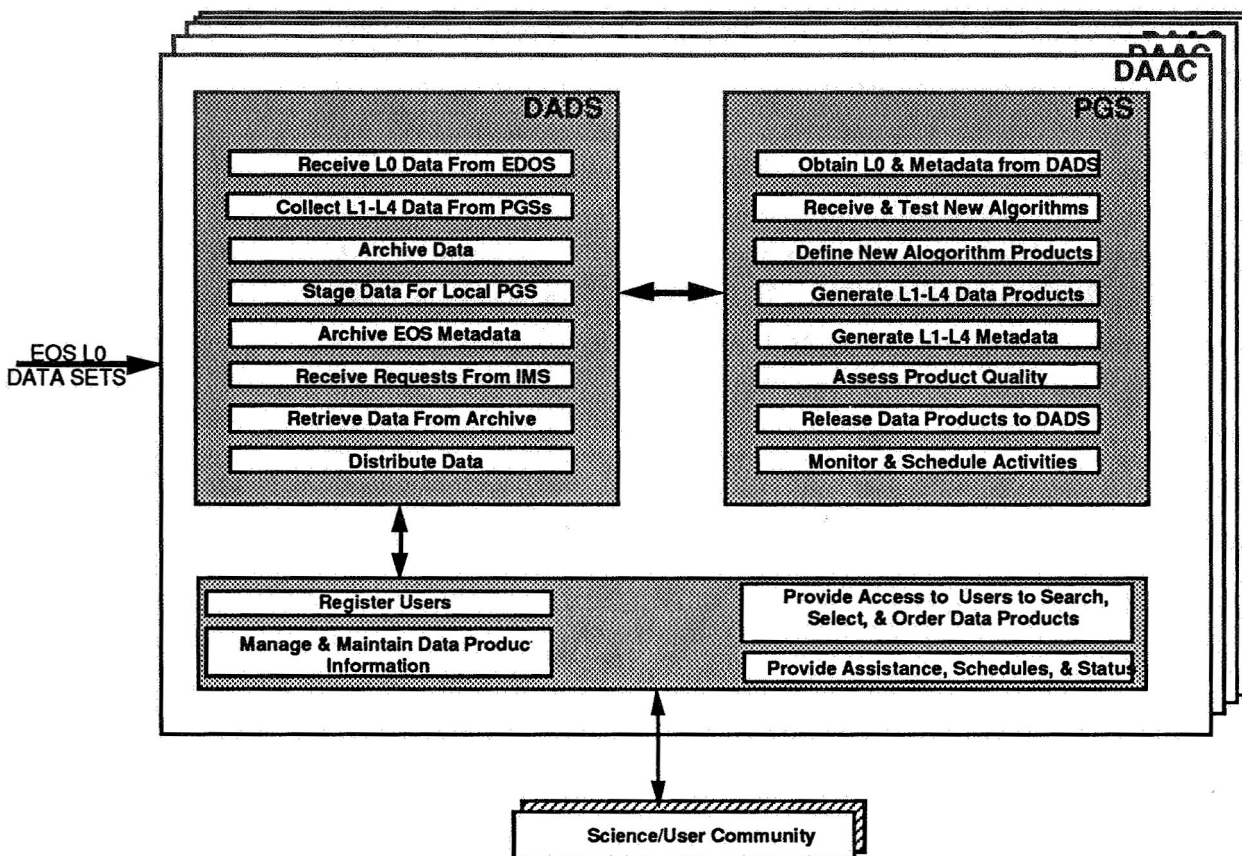


Fig. 4 — Science Data Processing

5. OUTLOOK

The ECS prime contract winner was announced in October after a lengthy procurement process. The award of contract is expected in January after completion of negotiations. The ECS contract is for ten years and will provide system design and development, system integration and testing, and operations and maintenance support. Another procurement process will begin shortly with the release of the EDOS Request for Proposal (RFP). THE EDOS is planned to be ready for operation a year before support for Landsat 7 is required. Similarly, an RFP for the EOS Ground System Independent Verification and Validation (IV&V) contract will be released soon.

6. ACRONYMS

CSMS	Communication & System Management Segment
DAAC	Distributed Active Archive Center
DADS	Data Archive & Distribution System
DIF	Data Interface Facility

Ecom	EOS Communications
ECS	EOSDIS Core System
EDOS	EOS Data & Operations System
EOC	EOS Operations Center
EOS	Earth Observing System
EOSDIS	EOS Data & Information System
ESN	EOSDIS Science Network
FDF	Flight Dynamics Facility
FOS	Flight Operations Segment
ICC	Instrument Control Center
IMS	Information Management System
IP	International Partner
NASA	National Air & Space Administration
Nascom	NASA Communications
NCC	Network Control Center
PDF	Production Data Facility
PGS	Product Generation System
SDPS	Science Data Processing Segment
SMC	System Management Center
SN	Space Network
TDRS	Tracking & Data Relay Satellite
TGT	TDRSS Ground Terminal

TELEMETRY GROUND SYSTEM DEVELOPMENT VIA THE COMPONENT APPROACH

N 9 4 - 2 3 9 4 9

Karen E. Thorn

Software and Automation Systems Branch, Code 522
Mission Operations and Data Systems Directorate
NASA/Goddard Space Flight Center
Greenbelt, Maryland 20771

ABSTRACT

NASA's deployment of major space projects such as the Earth Observing System (EOS) will demand increased functionality and ground-based telemetry processing performance well above current capabilities. At the NASA/Goddard Space Flight Center, custom hardware and software components have been developed and combined into a unique architecture to address this problem. The hardware components utilize Application Specific Integrated Circuits (ASICs) developed specifically to support NASA's telemetry data systems needs and designed to handle data rates up to 300 Mbps. A generalized set of software components, called the Telemetry Processing Control Environment facilitate the rapid construction of control and monitoring functions for the ground-based telemetry processing systems. This combination of hardware and software elements enables rapid construction of flexible, cost-effective telemetry processing systems capable of meeting the performance requirements facing NASA in the coming decade.

Key Words: Object-Oriented, VLSI, CCSDS, Telemetry, XWindows, C++

1. INTRODUCTION

The goal of today's spacecraft telemetry processing system developers is similar to the goal of early television pioneers. Technology has provided a revolutionary system for transmission of information. The goal is the very broad distribution of this technology to information providers and consumers. Three elements are required to meet this goal; they are the:

- development and adoption of standards to provide a framework of commonalty
 - development of engineering capabilities that meet these standards, and
 - development of low cost, easy to use, reliable products that apply these engineering advances.
- Development of standards in the area of space telemetry systems is well underway, meeting the first of these three elements. The second element is being

met through recent advancements in hardware and software technology. The final element is being met through the development of reliable products. This paper discusses the generic hardware components and the software components developed for ground-based telemetry processing system monitoring and control. The paper describes the telemetry data, stream ground-based processing, the problem incurred as telemetry processing requirements change, the component approach proposed to solve this problem, and future direction of the work in this area.

2. TELEMETRY DATA STREAM

Telemetry data stream processing can be broken into two elements-- execution, subdivided into pre- and post-execution, and data transfer. Pre-execution begins as spacecraft orbit and attitude data are provided to the Payload Operations Control Center (POCC). This data and the spacecraft's mission objectives are used at the POCC to produce command loads, sets of machine codes for operating the spacecraft. Post-execution, the science and engineering data collected and returned to the ground is processed and transmitted to scientists and engineers. Both pre- and post-execution data transfer is handled by an entity called the Space Ground Network. The Space Ground Network uses both ground-based communication links and domestic satellites for data transfer.

3. THE PROBLEM

The technology employed by today's Space Ground Network cannot handle the data rate and throughput requirements of tomorrow's complex missions. Many current ground-based telemetry processing systems are mini-computer-based with telemetry processing performed in software. This approach constrains data rate and throughput. The problem imposed by limited processing power can be illustrated by considering the data rate and throughput requirements of future missions that incorporate imaging technology. Very large volumes of data will be generated at very high rates. Both will quickly overwhelm current telemetry processing systems.

These functional and performance requirements are driving technology and development of future processing systems employing fast hardware components and embedded telemetry processing.

4. THE SOLUTION

4.1 HARDWARE

As part of an effort to reduce the costs of NASA ground-based data processing systems, a suite of generic hardware and software components has been developed by the Microelectronics Branch at GSFC to capture, process, and distribute spacecraft telemetry. In this effort, a functional component approach was adopted. The philosophy underlying this approach is to combine basic hardware elements into larger functional components that are easily configured into a low-cost, high-performance, high-reliability telemetry processing system. Two key elements supporting this approach are Very Large Scale Integrated (VLSI) circuits, and an advanced real-time software environment. These elements utilize telemetry distribution standards, such as Consultative Committee for Space Data Systems (CCSDS) standards, to build generic telemetry processors and in so doing, significantly reduce costs to only those incurred by hardware replication.

The VLSI circuits are the backbone of the functional component approach to developing low-cost, high performance telemetry processing systems. VLSI technology enables the processing systems to handle very high data rates, up to 20 Mbps, with future developments raising that rate to 300 Mbps. The VLSI Level Zero Processing (LZP) Prototype System is an example of the application of this approach to telemetry processing system development. The LZP architecture uses a functional components approach based on the Versa Module Eurocard Bus (VMEbus) with multiple microprocessors running concurrently. Telemetry data flows through a custom built pipeline where parallel processing supports the very high data rates.

Tightly coupled with the VLSI-based hardware is state-of-the-art real-time software technology. Together, these technologies allow the developers to design and implement highly functional, flexible data systems. Modular Environment for Data Systems (MEDS) is an example of an environment that helps to design data systems on a standard hardware platform. MEDS supports the basic software functions needed in all telemetry processing systems:

- setup application specific hardware and software,

- process telemetry data based on setup parameters,
- monitor the processing, and
- supply network support for remote operator interfaces and data transfer.

The infrastructure, provided by the real-time MEDS environment and the hardware allows timely design and construction of general purpose telemetry processing data systems. A control and monitoring environment for these generalized systems is easily implemented with a mapping between each hardware component and a corresponding control and monitoring software component. This control and monitoring is layered on the messaging scheme employed by all hardware components. In addition, a set of general purpose commands can be interpreted and executed by each custom designed component. (See Figure 1.1)

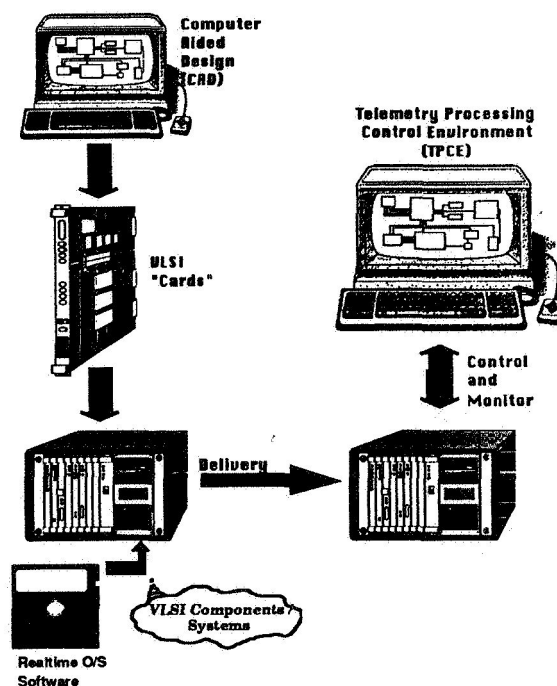


Figure 1.1 Building a Telemetry Processing System

4.2 SOFTWARE

The Telemetry Processing Control Environment (TPCE) provides a generalized suite of functions for control and monitoring. This suite allows the user to define processing actions to be taken by the hardware, to edit and store sets of hardware control information, to graphically and dynamically display telemetry processing status, to create and manipulate directories and catalogs of processing control parameters, to command the hardware, to establish

and manage network communications, and to manage the TPCE application itself. All communications between the TPCE and the MEDS-based hardware systems are done through the MEDS-defined protocol.

The TPCE was designed using object-oriented methodology and the software was implemented using C++, an object-oriented extension to the C programming language. The TPCE platform is a color graphics workstation based on the X Windows System and OSF/Motif. In selecting the platform, emphasis was placed on system usability and there was no requirement for real-time operating system environment. The user interface component of the TPCE system was generated with a User Interface Management System (UIMS) developed at NASA/GSFC called the Transportable Applications Environment Plus (TAE+).

The TPCE software allows the user access to any of eight TPCE subsystems described in the following paragraphs.

4.2.1 Communications

The communications subsystem is used to establish information exchange links between the workstation and the data system hardware. The communication package allows concurrent connection to multiple MEDS-based processing systems and manages the transmission of MEDS messages. The package uses TCP/IP sockets with network event notification provided by TAE+.

4.2.2 Subsystem Editor

The catalog editor allows users to specify complete or partial hardware configuration sets and to store those sets at the local workstation for later execution. Each configuration set, called a catalog, contains the specific setup definitions for each card to be configured.

4.3.2 Monitoring

The status monitoring subsystem handles the display of status information to the user. Status information for each of the hardware cards is gathered and displayed in a textual form. Future upgrades to the status monitoring capabilities will include graphical displays. The update rate for status display can be set by the user.

4.2.4 Control

The control subsystem allows the user to send commands to the hardware cards. This subsystem handles initialization, initiation, execution, and termination of a data system session. Typical

commands load setup data for a card, or enable a card using the setup data currently loaded, or reset data accounting statistics.

4.2.5 Event Log

The operator log provides a time-sequenced account of the actions taken by the user, the subsequent errors or warnings encountered, and any corrective measures that were taken in response to problems. Print and archival facilities for the log are also available. Developers and integrators of the data system can verify system configuration and monitor the installation, integration and execution.

4.2.6 User Preferences

The user preferences manager allows the user to tailor the interface to his/her satisfaction. Display panels can be rearranged via a window manager and foreground and background color schemes can be modified. The user has full control of the environment with which he/she is working.

4.2.7 Real-time Service

The real-time data subsystem allows the user to obtain Quick-look processed data, as well as real-time data packets, as the data is being processed through the system. All data is gathered and saved on the local workstation for future playback. Data accounting statistics, such as total packets with good quality data or with errors, are gathered and displayed to the user as the real-time data packets are processed.

4.2.8 Network Configuration

A network manager is the mechanism through which the network of MEDS-based systems is configured, i.e., the locations of specific MEDS-based system. The manager also monitors network activity, reporting problems to the TPCE for resolution, and allows the user to change the network configuration.

5. TYPICAL OPERATIONS SCENARIO

A typical scenario for TPCE system operation covers configuring, commanding, and monitoring the VLSI hardware. A TPCE user determines the type of information to be processed through the system, and configures the system hardware to process a specific type of information as required. The hardware components are configured using the specifications given in the corresponding catalogs. The necessary hardware configuration catalogs are then downloaded from the local workstation or loaded from local disk storage. Once the setup parameters have been

specified, the operator may establish a communications session (with the front end or "box") and initiate the telemetry processing session using those parameters. The data accountability for the current processing session is monitored via returned status. Real-time or post-session telemetry data products may be viewed.

6. PROCESSING SCENARIO

6.1.1 Spacecraft

Hardware components on-board the spacecraft package telemetry data into packets. Packets are encoded for data quality evaluation at ground sites. A fixed or variable number of packets are combined into frames with header information. This header contains information used at the various ground sites for time-ordering the packets and deleting redundant packets. These frames are combined into blocks that are used by the NASA Communications (NASCOM) Network for transport.

6.1.2 Data Processing

The reverse of the previous process is performed on the ground. The Frame Synchronizer Card performs synchronization on the frames to verify the correct ordering of data packets. Next the frames are transported through the Reed Solomon Decoder card that detects and corrects errors in the frames. The Packet Processor Card then processes each of the packet, combining them into datatakes. Datatakes are captured by the Data Capture Card and earmarked for distribution to various science and operations facilities. (See Figure 1.2)

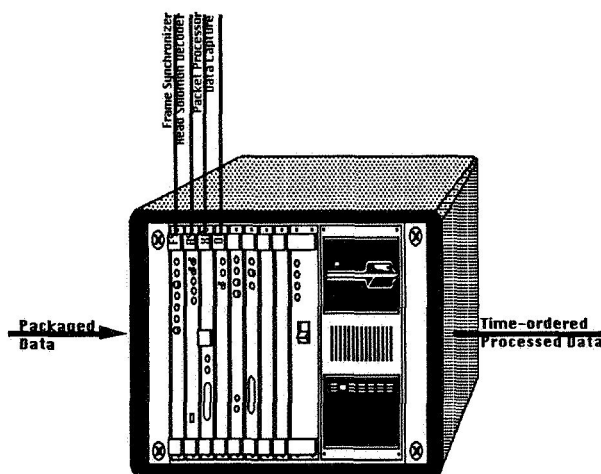


Figure 1.2 Typical Configuration including input and output products

7. TPCE SYSTEM DESIGN

The hardware components are object-oriented by nature (with similarities in configuration and actions):

- each card has a group of setup and status parameters.
- each card can accept and react to all the MEDS commands,
- each front end contains a group of cards,
- each front end can accept and react to all the MEDS commands.

This allows for a direct mapping of the control and monitoring software components to their hardware counterparts.

The top level view of a system is that of a front end within which the user may execute a data processing session. The communication paradigm remains the same across front ends, allowing each front end to handle the routing of commands to be processed to the hardware components. Each front end may be slightly different based on the cards in its configuration and may have additional actions the operator may invoke, such as retrieving real-time data or stored data. This view of the system allows each front end to be represented in the TPCE system as an object.

Each card is also represented as an object in the TPCE system. Generic commands, such as enabling and disabling, are accepted by every card. However, each card processes the commands differently in a manner appropriate to the purpose of the card. In the following paragraphs, italicized words are indicative of the software object representing the entity described.

Each *Card*, the software object representing the hardware card, has a configuration for setup and status data. The TPCE system provides for the configuration parameters to be stored in a file and loaded into the system upon startup. These configuration files contain the naming conventions for the parameters, the default values of those parameters, and the parameter's type. In addition, the *Card* structures have methods, a function on a particular software object, to read and write their appropriate data in forms compatible to the software's function.

One or more editors are associated with each card. *Editors* allow views of information to be presented to the user. The *StatusEditor* and *SetupEditor* for a

card present the status or setup data to the user in a graphical format, using the TAE panels for display. View files contain the mapping between the card parameters and their corresponding display entities. The maintenance of separate configuration data makes recoding unnecessary when changing the view of the data.

A *CardEditorManager* was developed to allow the creation a user-defined number of editors at start-up time. The environment set at startup specifies the number of each type of editor to be created. A pool of editors is maintained during application execution. As editors are requested they are removed from the pool of available editors, and are released when software processing is complete.

Card structures are also created and placed in a pool. This pool is managed by a *CardManager*. These card structures are assigned to the front end object as needed and returned to the pool after the session with the front end ends.

A *CatalogManager* allows the user to modify catalogs of Card setup parameters using a *SetupEditor*. The updated configuration may be saved to the workstation's local storage or downloaded for execution during a telemetry data session. The *CatalogManager* also uses the *Card* structure to save the parameters. The *CatalogManager* is responsible for local file storage and directory hierarchy traversal.

The communications segment of the TPCE software, the *IFMapper*, handles the interprets both incoming and outgoing messages. Outgoing messages are converted to a byte stream and are then sent to the *CommGate* for transmission upon command request from the operator. The *CommGate* handles the socket communications from the workstation to the front ends to which sessions have been established. Incoming messages are read as a byte stream and decomposed into the format used throughout the TPCE software. These messages are passed through the *IFMapper* for processing by the application.

These messages in internal format, called *MedsMsgs*, have two segments: a header containing information about the data to follow, and a body containing data. Subclasses of the *MedsMsg* class handle different types of messages, including status, command, catalog, and response messages.

8. ANALYSIS

The TPCE system and associated hardware have a number of advantages and disadvantages. The

functional approach to data system design is cost-effective from the hardware stand-point. The cost of building end-to-end data systems using this approach is replication cost instead of repeated design and construction cost. The adoption of standards and hardware designs based on standards have enabled repeated hardware replications. Processing is tailored to specific needs using hardware configuration specifications. The same data is used during the different processing stages, however each component extracts only relevant pieces.

The first advantage to the functional approach is enhanced capability. Future missions such as the Earth Observing System (EOS) will require telemetry processing systems to handle much higher data rates and volumes. Using prototyped VLSI components, a rate of up to 300 Mbps will be handled.

The second advantage is TPCE's modularity and flexibility, provided through an object-oriented design. In a prototyping environment where changes to software occur daily, the effect of software modifications can be localized. In addition, software can be reused. The functional component approach allows for easy tailoring of processing environments to the target mission.

An example of this localization of change can be seen in the design of the *Card*. Using the approach of editable files to hold Card configuration information provides the data system developers with an advantage of localizing the effects of hardware configuration changes. The front end hardware configuration can be changed on the fly without modifying the TPCE software. This also simplifies testing of the hardware components and integration testing of the completed system.

Another advantage to the TPCE system is fast execution. Using this editor pool scheme explained earlier, fast execution time can be achieved. There is, however, a trade-off between execution speed and higher memory usage, since all editors are maintained throughout the execution session.

Although there are many advantages to the current processing system, there are also disadvantages. One disadvantage is the MEDS-environment structure, the hardware's run-time environment. Word boundary overrides restricted workstation processing power. The volume of data accepted by the workstation posed processing power restrictions. These problems are being addressed in the next version of MEDS. For hardware components to be compatible with the

MEDS-based system, they must conform to the structure of the MEDS environment. Provision must be made for commanding and status monitoring using the MEDS system format. This requirement introduces some inefficiencies into the system, and network overhead during transmission.

Another disadvantage is the affect of an object-oriented methodology on execution time. The design imposes some structural inefficiencies affecting execution. For example, the use of *StatusEditor* setup files, while very efficient in terms of allowing changes to the configuration, introduces some overhead. Also, file management and memory usage overhead may be detrimental to some telemetry data processing applications.

A final disadvantage is TPCE's use of user interface. Any user interface will incur processing limitations, driving the execution speed of the application. The problems incurred will be addressed in the next version of the TPCE software.

9. SUMMARY

The TPCE system and the corresponding hardware it controls and monitors is the future in telemetry data system development. To fully develop the potential of future space missions, NASA's telemetry data systems must do more than simply meet specific technical requirements. They must provide reliable, low cost, modular systems which NASA and the user community can tailor in size and performance. These systems must allow growth and expansion in the years to come. Also, the push toward automated data driven operation of NASA's next generation telemetry data handling systems require standard system functional components that are virtually turnkey in operation. Even though these systems are a tightly integrated mix of hardware and software elements, they are but single elements in a large, highly complex Space Ground Network for telemetry data handling. The functional components approach and the Telemetry Processing Control Environment were designed to meet the processing needs of future missions.

10. REFERENCES

1. NASA/GSFC Code 521, 521-SRD-001, High Performance VLSI Telemetry Data Systems, September 1990.
2. NASA/GSFC, TAE Plus User Interface Developer's Guide Version 5.1, April 1991.

DECENTRALIZED ARCHITECTURE FOR DATA PROCESSING IN THE IN-ORBIT
INFRASTRUCTURE GROUND SEGMENT

P. Poirier
Matra Marconi Space France
Toulouse Cedex, France

*Paper presented but not submitted to proceedings
as of December 14, 1992.*

USING THE SPICE SYSTEM TO HELP PLAN AND INTERPRET SPACE SCIENCE OBSERVATIONS

N94-23950

Charles H. Acton, Jr.

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA USA

ABSTRACT

A portable multimission information system named SPICE is used to assemble, archive, and provide easy user access to viewing geometry and other ancillary information needed by space scientists to interpret observations of bodies within our solar system. The modular nature of this system lends it to use in planning such observations as well. With a successful proof of concept on Voyager, the SPICE system has been adapted to the Magellan, Galileo and Mars Observer missions, and to a variety of ground based operations. Adaptation of SPICE for Cassini and the Russian Mars 94/96 projects is underway, and work on Cassini will follow,

SPICE has been used to support observation planning for moving targets on the Hubble Space Telescope Project. Applications for SPICE on earth science, space physics and other astrophysics missions are under consideration.

Key Words: Data analysis, observation planning, ancillary data, geometry, Level 6 data

1. INTRODUCTION

Distributed information systems are becoming popular within the space science community. With affordable access to substantial processor power, large scale mass storage and high speed networks, scientists today wish to—or must—design and operate their own science data analysis systems.

Doing one's own thing often requires a substantial investment of local resources: repeated at each end-user site, this may seem unduly wasteful. Yet local processing offers many advantages and may be the only means for obtaining the derived products needed for one's scientific pursuits. Local processing affords the scientist or other end user a level of knowledge about the mechanics of the data processing often missing when the processing is largely done at a central site. With good in-house tools the user has specific knowledge about the algorithms and

processing parameters used and the processes applied as raw data (Level 0) are refined into useful products. More importantly, the user has the flexibility inherent in local control. Within some limits he may select what quantities are derived, which data are used in the process and when the processing takes place. The pipeline may be tuned or rearranged as data analysis requirements evolve or data sources change.

2. SPICE OVERVIEW

SPICE offers one approach for producing and using ancillary data within a distributed information system architecture. SPICE provides a portable, multi-discipline mechanism useful in both planning space science observations and in reducing the data obtained from those observations. Within the SPICE context, ancillary data are broadly defined as those used to determine:

- where an instrument was located while taking data,
- where the instrument was pointed and what targets it could see,
- how those targets would appear at the time of observation,
- how an instrument was acquiring data, and
- what else of significance to science data analysis was occurring.

In this context a "target" could be a whole planet, satellite, comet or asteroid, a surface or atmospheric feature, or a star or other galactic entity. Targets on planet earth are as valid as any others for application of SPICE technology.

2.1 SPICE Components

The principal components of the SPICE system are a set of elemental data files—called kernels—and software. Some of the SPICE software is used to produce kernel files, some is used to read those files to find information or calculate parameters needed to plan observations or interpret sensor data. The contents of the SPICE components are summarized in Figure 1. The reader can see that the acronym might better have been "SPICES."

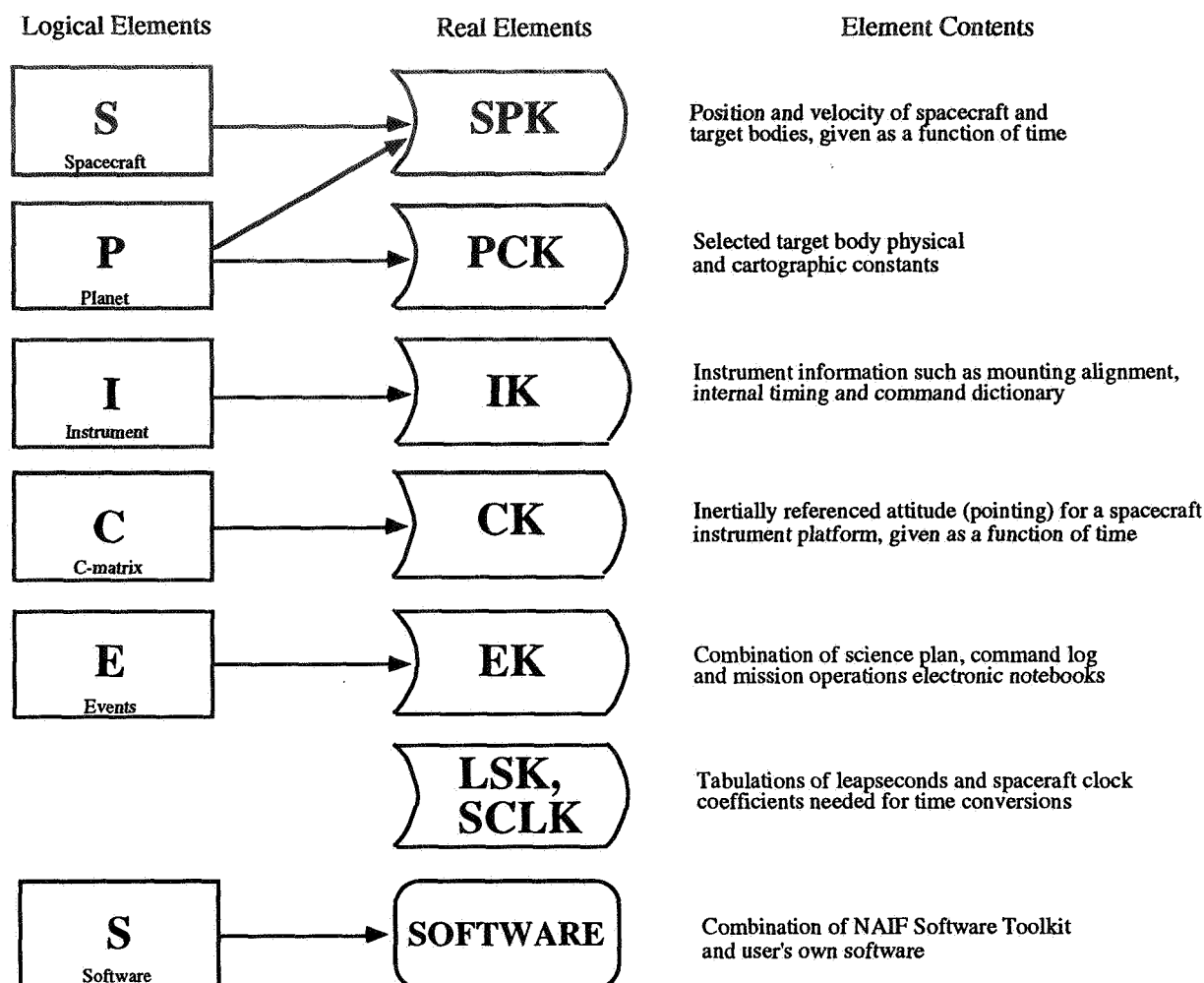


Figure 1. Principal Elements of the SPICE System

2.2 Making SPICE Kernel Files

Instances of each type of SPICE kernel file should be produced by the entity most qualified to do so—the person or group most familiar with the components and processes from which ancillary data are produced. For example, a project's navigation team responsible for estimating the orbit of a spacecraft would be responsible for producing SPK kernel files. Similarly a PI or facility instrument team would construct the I-kernel for that instrument (or would at least provide the information needed for constructing an I-kernel).

2.3 Distributing SPICE Products

Whatever the source, SPICE kernel files and NAIF Toolkit software are normally collected and cataloged in a central repository for subsequent distribution to

end users. For a flight project the repository likely resides at the mission operations center for the duration of the project. In other cases the repository could be a local, national or international archive center established for the archive and distribution of discipline specific science data products.

A mission operations center might make automatic, regular distributions of project SPICE data to individuals or agencies directly associated with the project. Alternatively, an end user needing ancillary data could contact a mission or discipline archive center to order specific data of interest. The order might be effected through an on-line catalog, or it could be placed with the archive's staff using electronic mail or telephone.

SPICE data are delivered to the user by whatever means is available and best suited to both parties.

Magnetic tapes or diskettes are the traditional distribution media; these are still sometimes the best or only method available. Worldwide electronic networks offer convenience and immediacy for small to modest data volumes. CD-ROM and write once CD (WOCD) will soon play a major role.

2.4 Using SPICE Kernels and Toolkit Software

The scientist or engineer builds an application program to address a particular need. This program integrates appropriate subroutines from SPICELIB (SPICE Library)—the principal component of the NAIF Toolkit software—with the user's own subroutines. Some SPICELIB routines provide access to the data within SPICE kernel files. For example, calling SPICELIB routine SPKEZ returns the Cartesian state vector giving the position of a target relative to an observer. The “observer” is typically a spacecraft, but it could be any object for which ephemeris data is available within an SPK kernel file.

Other SPICELIB routines combine data extracted from the kernels to produce derived (higher level) geometric parameters or related ancillary information needed to plan observations or to interpret science data. Spacecraft altitude, lighting angles and latitude and longitude of an instrument's optical axis intercept with the target are typical of the computations available within SPICELIB.

3. INTENDED CHARACTERISTICS OF THE SPICE SYSTEM

3.1 Correct Results Are Obtained

It is imperative that, within realistic limits, an ancillary information system such as SPICE provide results that are “correct”. Whether or not a SPICE user gets the intended results depends on several factors.

- Correct and accurate data must be assembled for placement in SPICE kernel files.
- Software used to produce SPICE kernels and subsequently to read them and compute derived quantities must be accurately specified and must perform as specified.
- SPICE kernel files must be correctly labeled.
- Customers—kernel producers as well as kernel users—must be able to determine with confidence which SPICELIB subroutines to use to yield the intended results. Customers must also realize when a desired functionality is not (directly) available within the SPICE library.

Computation precision and methods of handling exceptional cases are overarching factors that add another dimension to the issue of determining correctness.

Seeking out substantial, definitive customer feedback seems the best method of obtaining a composite measure of correctness covering the factors above. In this regard the SPICE system—in its current state—is not without shortcomings. Nevertheless, most users report success with a wide range of SPICE applications. The growing number of repeat customers and referrals further bolsters the sense of success.

3.2 Reasonably Easy to Use

Ideally all software would be easily used by experts and novices. But to a certain extent one must trade off simplicity against functionality; the easier a program is to use the more difficult it becomes to have it accomplish significant tasks. Similar tradeoffs are made between ease of use versus portability and broad applicability.

How easy it is to use a piece of software is very subjective. It would be misleading to assert that SPICE is easy to use. But it may be said that a substantial fraction of the NASA resources invested in SPICE development have been applied to documentation and tutorial materials created for end users, and these have helped the more than 200 SPICE users achieve their objectives.

3.3 Portable Data and Software

Early information systems were centralized—accessed by users who came to the mission operations center or who dialed in from a remote location. Portability was not an issue. More recently, data have been distributed with a file description document; it is the user's responsibility to decipher that document in order to write code to read a dataset. Now the trend is towards increasingly decentralized systems, with both data and allied software delivered to users operating on many different types of computing platforms.

Portability of SPICE kernel files is achieved by using text-format files. Several of the kernel files exist only in text format and so are readily ported. Those kernel files that are normally in binary format are translated to text using a portable NAIF Toolkit utility program prior to being ported to a heterogeneous host. The same utility program is used to translate the file back to the binary format of the new host once the data have been moved there. This translation process takes a little extra time and transfers involve more data. But the scheme works across the many computer systems and networks in use by scientists and engineers.

Achieving software portability is an elusive goal, traded against robustness and ease of use, and achievable in only a marginal sense. SPICE uses ANSI FORTRAN 77 code, and while environmental dependencies cannot be totally eliminated, in SPICE they have been minimized, isolated and clearly

documented. Changes needed to port the NAIF Toolkit software to several popular platform/compiler combinations are implemented and tested before the Toolkit is released to the user community; the current complement is listed in Table 1.

Platforms To Which NAIF Staff Have Ported The Toolkit	Platforms To Which Others Have Ported the Toolkit
DEC VAX VMS Hewlett Packard 700 Sun PC/Lahey EMS-32 FORTRAN Macintosh/Language Systems FORTRAN NeXT/Absoft FORTRAN	DEC VAX Ultrix Cray Data General Silicon Graphics (IRIS) IBM mainframe

Table 1. Examples of Platforms Where NAIF Software Is Operational

Several customers have inquired about the availability of the NAIF Toolkit in the C language—it is not. However, many customers successfully call SPICELIB routines from C programs. Most compilers support cross-language compatibility.

3.4 Flexibility - Separable and Extensible Components

To endure, a data system must be easily adapted to meet new or revised requirements.

While the SPICE kernels and the allied Toolkit software are designed to operate as an integrated system, subsets of SPICE also prove useful. One example of SPICE component separability can be found in the widespread use of SPK kernel files containing only planet and satellite ephemeris data (no spacecraft ephemeris) for terrestrial observation planning and data analyses.

Each kernel design had extensibility as a major objective. Two approaches were used. For those kernels that are text files, data are formatted in a simple but flexible **KEYWORD = VALUE** style. New keyword/value pairs can be added as needed. Descriptive text that is ignored by kernel access software can be inserted wherever appropriate.

The binary kernels are built upon specially developed data structures that were designed to accommodate growth. Each structure supports a family of data types, unlimited in number. As an example, the SPK kernel file family currently consists of eight data types, such as Chebyshev polynomials, conic elements and discrete state vectors.

3.5 Affordable

Economic considerations—always important—today receive extra attention from NASA managers. SPICE rates high in this regard, partly because of the flexibility and wide applicability noted earlier. These qualities allow the cost of system development and maintenance to be shared across a broad user base.

Equally important is the attention paid to ensuring that SPICE software maintenance costs will be reasonable. The subroutines are written in a consistent, easy-to-read style and are highly annotated—approximately 70 percent of the source code is documentation. The NAIF staff refrains from using FORTRAN features that frequently make code difficult to maintain: examples are COMMON blocks and EQUIVALENCE statements. But use of a few extensions to the ANSI FORTRAN 77 standard which promote code maintainability are permitted in the local master copy of the Toolkit. A precompiler is used to translate this code into pure ANSI FORTRAN 77 before Toolkit deliveries are made.

4. SPICE KERNEL FILE SPECIFICATIONS

4.1 SP Kernel File

SPK files contain data that, when interpreted by SPK "reader" subroutines, yield position and velocity of one "ephemeris object" relative to another at the time requested by the user. In the language of SPICE, spacecraft, planets, satellites, planet-satellite barycenters, comets, asteroids, the sun and the solar system barycenter are all ephemeris objects.

The position and velocity returned from SPK readers at the requested time are given as Cartesian vectors in a user-selected inertial reference frame, such as B1950 or J2000. Either true or apparent position may be requested. SPK kernel files are structured as binary, direct-access files to provide quick response to requests.

4.2 PC Kernel File

PCK files contain the names and values of physical and cartographic constants for target bodies—items that change infrequently. NAIF assembles and distributes a version of the PCK file that contains those data needed to determine spin axis orientation, size and shape (three axes) and location of the prime meridian for all planets and satellites, as determined by the International Astronomical Union. A PCK file could include similar kinds of target model data for comets or asteroids. Data in PCK files are most frequently used to define a body-fixed reference frame used to compute locations of objects in cartographic coordinates (latitude and longitude).

All PCK files use text format, so it is easy for users to change parameter values, add or delete data items, or add or delete target bodies in their local copies. Any text editor will do.

Data in a PCK file are formatted as KEYWORD=VALUE structures. Notes or references in free format text may be inserted where desired throughout the file. (Only the KEYWORD=VALUE data are read by Toolkit software.)

4.3 I Kernel File

IK files contain an assortment of science instrument-related information. A separate IK file is constructed for each instrument. These files are text format, using the same KEYWORD=VALUE structure used in the PCK file.

An IK file contains an instrument's mounting alignment relative to a spacecraft reference frame for which a CK file exists. An IK could include data that model articulation of an internal mirror used to point the instrument's optic axis. Also possibly found in IK files are an instrument's internal timing offset parameters, used to correlate data time tags with the instant that photons or particles impinged on the instrument's sensor.

An instrument's command dictionary, operating description and references to calibration files and pertinent instrument documents should be included in its IK file. Such text fields are not accessed by

Toolkit software, but may be easily displayed or printed using standard computer utilities.

4.4 C Kernel File

CK files consist of time tagged quaternions used by Toolkit "readers" to construct the transformation matrix that rotates a vector from an inertial reference frame to coordinates fixed to an instrument platform. The data in a CK kernel file are derived from a spacecraft's attitude and articulation telemetry data.

CK kernel files use the same fundamental structure found in SPK files and offer the same flexibility of data content and quick data access. At present three CK data types are supported.

4.5 E Kernel File

The E-kernel consists of three distinct components: Science Plan, Events and Notebook. The Science Plan and Notebook components use identical text file structures. The Events component is a binary file, somewhat like the SPK and CK files, although the underlying data structure is different.

Time and "instrument" ID are the parameters common to all three EK components. Here, "instrument" is used in a broad sense: the spacecraft, the ground data system and any other mission element of interest may be treated as an EK "instrument" along with the bona-fide science data sensors.

Each science instrument team and mission operations team is encouraged to supply inputs to each of the EK components. The collection of inputs for each EK component type are merged in daily or weekly collections.

The Science Plan component records broad scientific objectives for the operation of an instrument over the applicable time period.

The Events component records the commands—specifically, English language transliterations—sent to the spacecraft and its instruments in order to carry out the objectives of the Science Plan.

The Notebook component is used to record notes about problems, unexpected situations or general information of interest observed by scientists and mission operations staff as the instrument and spacecraft commands get executed. In an ideal world where everything worked exactly as planned, the Notebook component might never be used.

4.6 Miscellaneous Kernel Files

Two miscellaneous kernel files are needed to support typical planetary science missions. One, the Leapseconds Kernel (LSK), contains a tabulation of the leapseconds and related data needed by SPICELIB subroutines used to convert between civil time (UTC) and ephemeris time (ET, also called Barycentric Dynamical Time or TDB). This conversion is needed because ET is the independent variable used in SPICE SPK and CK files. (It could also be used in EK files.)

Similarly, the Spacecraft Clock Kernel file (SCLK) contains a tabulation of data needed for conversion between ET and the spacecraft clock time tags assigned to science and engineering telemetry returned from the spacecraft.

4.7 Kernel File Labeling

All SPICE kernel files contain an area set aside for internal labels. These internal labels can be used to help construct an SFDU, PDS Label, FITS Label or other formal metadata structure required by a particular discipline information system.

5. NAIF TOOLKIT CONTENTS

The principal component of the NAIF Toolkit is a library of ANSI FORTRAN 77 subroutines and functions named "SPICELIB." Users of SPICE kernel files usually need little or no understanding of

the file structures and specific contents since their interface to these datasets is always realized through the SPICE kernel file "reader" subroutines found in SPICELIB.

In addition to the kernel file readers, SPICELIB contains a broad set of subroutines designed to further assist scientists with the planning or interpretation of space science observations. Examples of the functional categories addressed by these modules are time translation, reference frame and coordinate conversion, solid geometry, vector and matrix algebra, conic element applications, string manipulation, parsing and array manipulation.

Beyond SPICELIB, the Toolkit contains "cookbook programs" that are samples of typical use of SPICELIB subroutines and functions. It also includes several important utility programs that provide conversion, summarizing and labeling functions for binary SPICE kernel files.

6. SPICE SYSTEM APPLICATIONS

The SPICE system has been built and tested in a predominantly planetary science environment; its use in supporting science observation planning and science data analyses in conjunction with planetary flight projects is well established. SPICE has also been applied in other space science disciplines. Table 2 summarizes current and planned applications. It also identifies examples of possible further use.

Current Applications	Tentative or Under Discussion	Examples of Future Possibilities
Voyager (Uranus/Neptune) Magellan Galileo Hubble Space Telescope Mars Observer Mars 94/96 Cassini	Radioastron/VSOP Spectrum-R/G Clementine	Eos Interbol Discovery Program Lunar Scout SIRTF AXAF Pluto Flyby

Table 2. Current and Possible SPICE Applications

SPICE is also used informally to support a variety of other programs and applications; IUE observations of satellites, terrestrial observatory observations, mission concept studies and solar system dynamics analyses are among these.

With each new application the growing cadre of scientists and engineers already familiar with SPICE will find it easy to deal with processes requiring the use of ancillary data. And with each new SPICE

application funding organizations will find the marginal cost for adapting and operating this proven information system to be very reasonable.

This research was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

AN ENGINEERING DATABASE MANAGEMENT SYSTEM FOR SPACECRAFT OPERATIONS

N 94 - 23951

Gregorio Cipollone, Michael H. McKay

Missions Operations Department, ESOC
Robert-Bosch Strasse 5, W-6100 DARMSTADT, GERMANY

Joseph Paris

BIM
Kwikstraat 4, B-3078 EVERBERG, BELGIUM

ABSTRACT

Studies at ESOC have demonstrated the feasibility of a flexible and powerful Engineering Database Management System in support for spacecraft operations documentation.

The objectives set out were three-fold: first an analysis of the problems encountered by the Operations team in obtaining and managing operations documents; secondly, the definition of a concept for operations documentation and the implementation of prototype to prove the feasibility of the concept; and thirdly, definition of standards and protocols required for the exchange of data between the top-level partners in a satellite project.

The EDMS prototype was populated with ERS-1 satellite design data and has been used by the operations team at ESOC to gather operational experience.

An operational EDMS would be implemented at the satellite prime contractor's site as a common database for all technical information surrounding a project and would be accessible by the co-contractor's and ESA teams.

Keywords: operations, documentation, spacecraft, database

1. BACKGROUND

During the design of a satellite and after its manufacturing, a large amount of information has to be transferred from the prime contractor to the European Space Agency (ESA). The most important document is the Flight Operations Manual (FOM).

Also, to respond to the needs of Operations and Assembly, Integration and Validation (AIV), ad-hoc documentation has to be produced.

All this documentation is delivered in a paper-based form. Thus, once this documentation is received by the operations teams, the information contained therein needs to be retyped to produce e.g. the Flight Operations Plan (FOP) - the document which contains all the nominal and contingency flight procedures for a space mission.

This effort requires enormous manpower from the operations teams. Also, production of operations documentation by spacecraft contractors historically receives a lower priority.

2. PROBLEM

The first consequence of this situation is that FOM's - these are produced by spacecraft sub-system - arrive late e.g. after the launch of the spacecraft.

Furthermore, due to the intrinsic complexity of current generation spacecraft systems, problems arise in the overall management of the documentation. Thus, the spacecraft documentation will contain incoherent information. Also, the lack of management in spacecraft design changes has impacts on mission control.

The nature of the documentation delivery - paper - generates huge distribution costs e.g. copying, mailing.

Finally, because the documentation is not stored in a single repository, there is no complete access to information.

3. ANALYSIS

One can identify a potential risk of mission failure (or at least a risk of reduced mission capability) from the above problems. Hence, the requirement to analyse the operations documentation production and review cycle within a spacecraft mission.

Analysis of the ERS-1 mission (as a case study of a typical current generation mission) has revealed several shortcomings in all of the aspects of operations related documentation: generation, distribution, maintenance, review and archiving. The analysis included the working procedures of prime and co-contractors as well as the operations and spacecraft check-out teams.

The main shortcoming identified was the lack of a single (and complete) repository for all operations documentation for a space mission. A concept for an Engineering Database Management System (EDMS) for spacecraft operations was thus developed.

4. DEFINITION OF EDMS CONCEPT

A set of constraints - technical and operational - dictated the main aspects of the EDMS concept.

4.1.1. Distributed architecture

Due to the distributed nature of spacecraft and mission development it is essential that each partner (prime, co-contractor, operations, etc..) in a spacecraft project can work independently on their own documentation and yet still have access to a common repository of information. A network of UNIX workstations and servers would satisfy this constraint.

4.1.2. Open systems

Because of the large number of partners involved, one must build an EDMS on open systems. This includes both hardware and software solutions. Software solutions must place the emphasis on neutral formats and standards for data exchange (e.g. SGML for text) and less on selection of specific software packages.

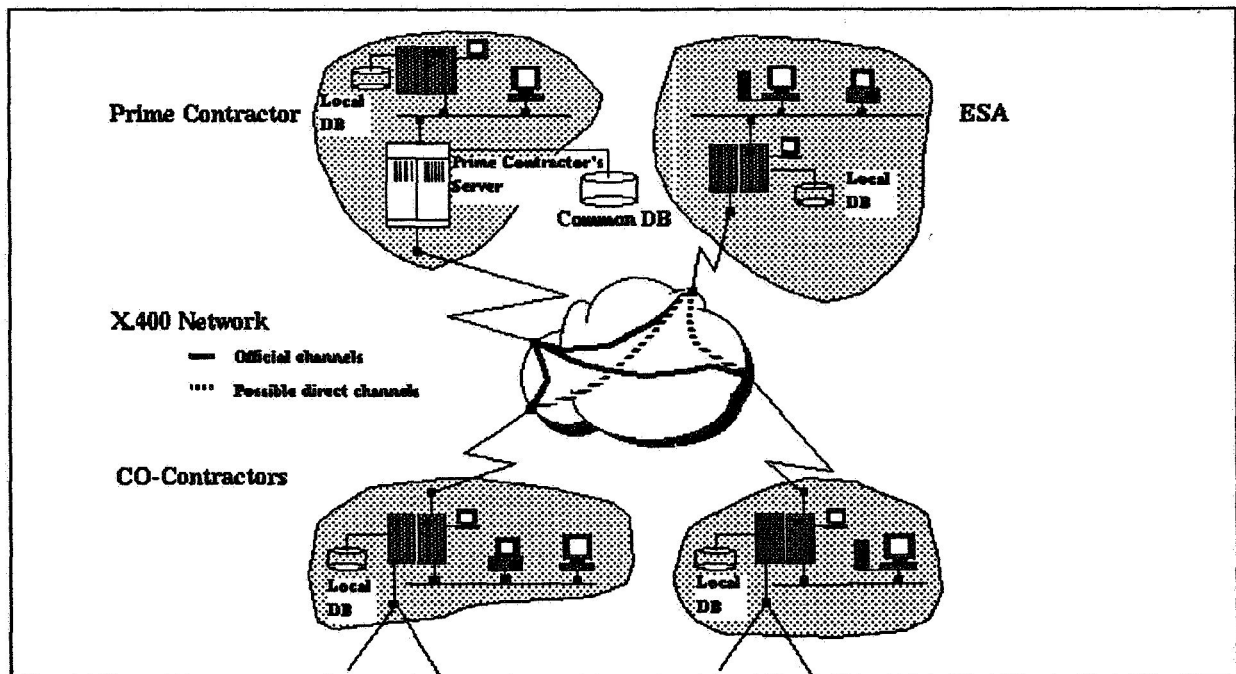


Fig. 1: Proposed network architecture for an operational EDMS

4.1. Constraints

4.1.3. Wide area networking

Since a spacecraft project involves a large number of partners, typically spread-out over several countries, an efficient means of distributing large volumes of documentation is needed. A Wide Area Network (WAN) based on standard communications protocol would meet this requirement.

4.1.4. Operations-friendly environment

Use of an EDMS and production of operations documentation must follow closely procedures used in current paper-based space missions e.g. document review cycles, production of Review Item Discrepancies (RID's), Non-Compliance Reports (NCR's), etc..

Furthermore, the types of current documentation must be maintained e.g. User Requirements Documents (URD's), System Requirements Documents (SRD's), Interface Control Documents (ICD's), FOM's, etc..

4.2. Functionality

Functional requirements for the EDMS generated from the above problem analysis and constraints.

4.2.1. Desktop publishing

Requirements here call for non-destructive editing (annotations), multi-author access, hyper-media functions, version and configuration control.

4.2.2. Communications

Identified requirements include E-mail, dissemination of documentation, delivery acknowledgement, access and security.

4.2.3. Data archiving and retrieval

Main requirements defined include adoption of unified data model and neutral data format for information storage and exchange, document browsing and navigation, text and concept retrieval.

4.3. Responsibilities

The mission prime contractor is seen as the key role player in any implementation of an EDMS. The prime contractor will be responsible for the gathering, maintenance and distribution of documentation from a Central Database (CDB).

Co-contractors, ESA Project Offices, mission operations and check-out teams would maintain their own Local Database (LDB) which could contain an up-to-date version of the CDB as well as their own 'working' copy.

4.4. Usage Scenarios

Usage scenarios would map the procedures from conventional project review cycles e.g. Baseline Design Reviews (BDR's), System Requirement Reviews (SRR's), FOM reviews, etc.. Additionally, the format and structure of project documents would be maintained in support of these formal reviews.

5. IMPACT ON OPERATIONS

The foreseen implementation of an operational EDMS will have several positive impacts on spacecraft missions:

- Increased productivity - within ESA as well as the contractors
- Improved communication of information
- Reuse of information
- Reliability and manageability of information

In addition to the improved productivity within the operations teams in preparing documentation, it is anticipated that there will be a minimisation of response time in the case of a spacecraft anomaly.

6. PROTOTYPE IMPLEMENTATION

A prototype was implemented, containing the main functions of an operational EDMS, in order to demonstrate the feasibility of the concept described above.

The main elements of the prototype included:

- ♦ Sun workstations
- ♦ Commercial-Off-The-Shelf (COTS) S/W packages:
 - DTP tool, *Framemaker*
 - RDMS tool, *Oracle*
 - Information retrieval tool, *Topic*
 - DB front-end/MMI tool, *CIM-Linc ID*

- X. 400 messaging system, *BiM-Mail*

They must also be able to accommodate the various constraints pertaining to specific aerospace domain

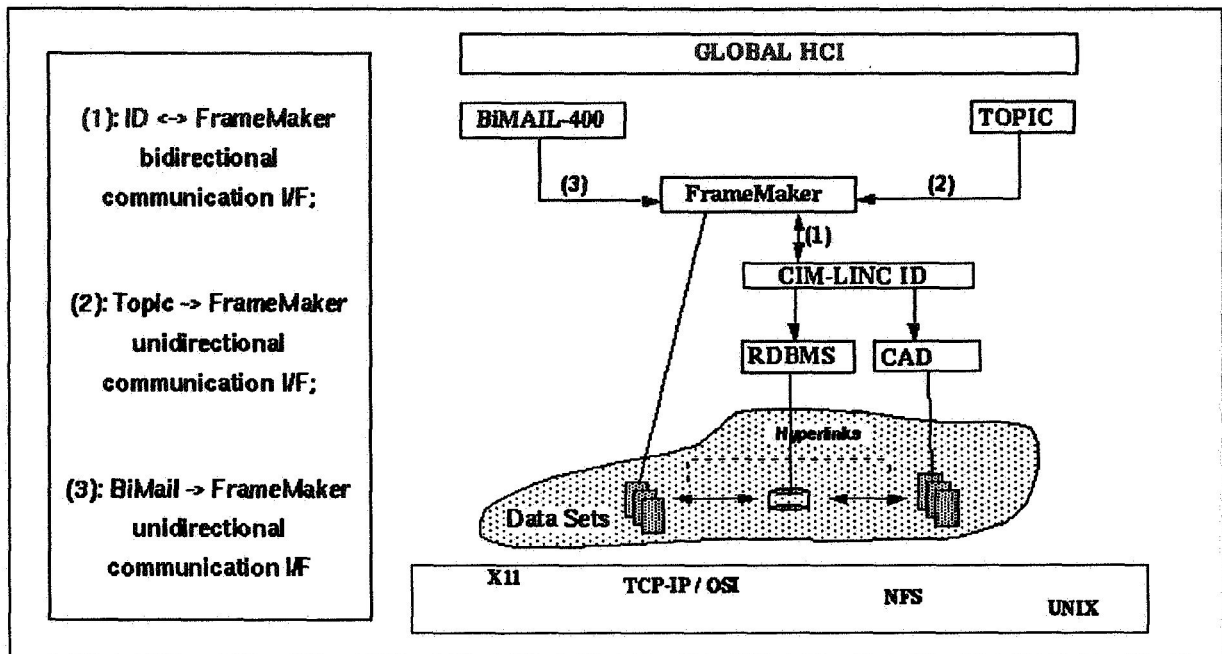


Fig. 2: Software architecture of EDMS prototype

The prototype was populated with data from ERS-1 platform and the IDHT instrument. A review scenario was developed for the IDHT FOM life-cycle.

7. PROTOTYPE DEMONSTRATION

The prototype was demonstrated at ESOC in February 1992. The demonstration consisted of two workstations at ESOC, Germany - acting as the operations team - linked to one workstation based at Everberg, Belgium - acting as the prime contractor. The link was provided by ESA's X. 25 network, *ESANET*.

The prototype has since then been delivered to ESOC where it has been evaluated by the operations team.

8. STANDARDS

The adoption and use of neutral standards for the representation and the exchange of design knowledge is a key factor for the successful deployment of EDMS in real space projects. These standards have to rely on existing international recommendations when available and applicable.

in terms of accuracy, flexibility and confidentiality.

Current study effort has already identified Standard General Mark-up Language (SGML) as a potential standard for text exchange. Above and beyond this, the adequate definition of related Document Type Definitions (DTD's) is required - to ensure information completeness - for all documentation generated during a space mission life-cycle.

9. CONCLUSIONS

The feasibility of an operational EDMS has been proven through the implementation and demonstration of the prototype.

The prototype demonstration received a positive response from major potential users of an EDMS: prime contractor, operations and check-out teams.

There is a strong interest from spacecraft prime contractors to get involved in a pilot implementation of an EDMS within in a real space project.

The ESIS Query Environment Pilot Project **N 9 4 - 2 3 9 5 2**

Jens J. Fuchs¹ Alessandro Ciarlo² Stefano Benso³

¹Computer Resources International A/S
Bregnerødvej 144, DK-3460 Denmark
Phone: +45 45822100

²ESA/ESRIN ISD
Via Galileo Galilei
I-00044 Frascati, Italy
Phone: +39 6 941801

³CISSET Space Division
Via Salaria 1027
I-00138 Roma, Italy
Phone +39 6 881701

1. Introduction

The European Space Information System (ESIS) was originally conceived to provide the European space science community with simple and efficient access to space data archives, facilities with which to examine and analyze the retrieved data, and general information services. To achieve that ESIS will provide the scientists with a discipline specific environment for querying in a uniform and transparent manner data stored in geographically dispersed archives. Furthermore it will provide discipline specific tools for displaying and analyzing the retrieved data.

The central concept of ESIS is to achieve a more efficient and wider usage of space scientific data, while maintaining the physical archives at the institutions which created them, and has the best background for ensuring and maintaining the scientific validity and interest of the data. In addition to coping with the physical distribution of data, ESIS is to manage also the heterogeneity of the individual archives' data models, formats and data base management systems. Thus the ESIS system shall appear to the user as a single database, while it does in fact consist of a collection of dispersed and locally managed databases and data archives.

The work reported in this paper is one of the results of the ESIS Pilot Project which is to be completed in 1993. More specifically it presents the pilot ESIS Query Environment (ESIS QE) system which forms the data retrieval and data dissemination axis of the ESIS system. The others are formed by the ESIS Correlation

Environment (ESIS CE) and the ESIS Information Services.

The ESIS QE Pilot Project is carried out for the European Space Agency's Research and Information center, ESRIN, by a Consortium consisting of Computer Resources International, Denmark, CISSET S.p.a, Italy, the University of Strasbourg, France and the Rutherford Appleton Laboratories in the U.K. Furthermore numerous scientists both within ESA and space science community in Europe have been involved in defining the core concepts of the ESIS system.

2. The ESIS Query Environment

Past and future satellite missions have and will generate massive amounts of data which must be archived and indexed for subsequent scientific processing and analysis. Together with bibliographic databases and object catalogues these data archives represents an enormous and very valuable source of scientific information. However, the efficient use of these data banks are today obstructed by a number of factors:

- o The structures and tools available for querying and working with a given database are often highly mission-dependent, and not standardized.
- o Database management systems and the query languages provided are different making it difficult to learn how to use new database systems.
- o Data in different databases cannot easily be combined.

- o Instruments may have different sensitivity, filters, resolution etc.
- o A number of different data exchange formats exist for transfer of data to users equipment.

As the cost of a satellite mission is very high, it is highly desirable to be able to make efficient use of data already collected thereby enabling future satellite missions to have more specific goals, supplementing data already collected, or providing necessary additional information on certain objects.

The initial goal of the ESIS QE is to overcome the mentioned obstacles, by providing researchers with an efficient tool for access to and use of space scientific data. Furthermore the project shall provide future missions with a structured environment for making their data available to researchers. This will include standard formats, structures and query languages.

2.1 Interoperability of the ESIS QE

Providing common access to two or more databases which use different DBMS can be achieved either by trying to get them to work transparently together or by providing higher or lower degrees of interoperability.

The lowest level of interoperability may be achieved by a basic system containing a directory in which the user can identify the archives containing specific information, and a communication infrastructure allowing remote access to the archives. The next levels are characterized by improved communication facilities providing automatic logon to remote hosts, and by the introduction of common query languages which can be used with the archive specific datamodels, and are interpreted by front end servers of the specific archives. The highest levels of interoperability introduces a common data model or schema, but may explicitly fail to address all the problems of syntactic and semantic consistency which must in principle be solved before the complete integration is achieved.

Within the ESIS QE Pilot Project the original aim was to achieve a very high level of interoperability through the provision of a common schema within each relevant scientific discipline, and a mapping mechanism allowing these common schema to be mapped physically and semantically on the underlying archives.

2.2 The Scientific Disciplines

In the pilot project two scientific domains were selected, namely:

- o Astronomy and Astrophysics
- o Space Physics

Within astronomy the following archives were selected for integration into the ESIS QE system:

- o IUE: (Infrared Ultraviolet Explorer, Villafranca Tracking Station, Spain)
- o STARCAT: (ESO, Garching, Germany)
- o EXOSAT: (ESTEC, Noordwijk, The Netherlands)
- o SIMBAD: (Strasbourg Observatory, France)

For Space Physics, only one archive was selected, namely:

- o GDF: (Geophysical Data Facility, RAL, United Kingdom).

Finally both domains were to be supported in the area of bibliographic data by providing access to the ESA-IRS database system operated at ESRIN.

It is foreseen that users will have varying degrees of computer literacy and database experience. It is therefore considered extremely important that the ESIS QE can cater to the occasional user, requiring help and assistance, as well as to the experienced user who may have his/her favorite database environment.

2.3 The QE Functions

The ESIS QE system provides the following main functionalities:

- o A user interface supporting the formulation of queries in various ways.
- o For each domain a common conceptual data model, defining the entities and attributes available for querying.
- o The translation of the query expressed relative to the conceptual model into a set of queries targeted against one or more archives, i.e. physical data-

models.

- o The retrieval and integration of the physical data from the various archives, and the forwarding and presentation of these data back to the user.
- o The provision of a user specific data repository, the Personal Database, allowing the user to maintain a collection of retrieved data for subsequent correlation and refinement.
- o Facilities for import and export of data allowing the retrieved data to be analyzed further using tools of the individual scientist or e.g. the ESIS CE.

3. The ESIS QE Architecture

The ESIS QE system is a distributed system by nature. It must provide a geographically dispersed user community with access to an almost equally dispersed set of archives and databases in the most effective and cost efficient manner. To achieve this a layered client/server approach has been adopted such that conceptually the system can be viewed as being composed of three logical components and the physical Archive Layer, which work together transparently.

Query Agent component (QA)

Provides the user interface and query formulation and result presentation facilities based on the common conceptual model. Provides service functions and interfaces to other systems, e.g. the Correlation Environment.

Query Executer component

Collects the user queries identifies the potential archives which may be sources of data for the specified query. Generates a set of archive specific queries and forwards these to the next layer. Upstream it collects results from the archives, integrates them and forwards them to the user.

Local Query Server component

Translates the archive specific query into the local archive query language and submits it for execution. Retrieves the results and homogenizes it depending on data type and use. Applies special functions not directly available in the archives on the temporary results and forwards the result to the Query Executer component.

Host Archive

The host database system which contains data such as catalogues of astronomical objects, mission logs, data sets, bibliographic references, general information about the scientific community.

The software components constituting the three top layers will be distributed among three physical platforms: the User Platform, the Access Point and the Archive Node. The User Platform hosts software responsible for the Query Agent functionality, the Access Point the Query Executer and the Archive Node the Local Query Server functionality. However it is foreseen that the same physical machine will host modules overlapping the conceptual components.

3.1 The Pilot System Architecture

Figure 1 provides an overview of the overall pilot system architecture. The hardware architecture is composed of several distributed computers interconnected via local area networks and wide area telecommunication networks. The constituents of this architecture are:

- o The User Platform, which are the work stations used by the scientific user to access the ESIS system, and do local data processing. A User Platform will be located at the scientists institution, interconnected via a network to an Access Point. The ESIS QE pilot system will support three different types of workstations: VAX/VMS with DECWindows Motif, SUN/UNIX with OSF/Motif.
- o The Access Point, is a VMS based machine which provides the first layer of server capability, and the further access to the Archive Nodes. In the pilot configuration there will be only one Access Point, located at ESRIN. However the overall concept is to have distributed Access Points, where each will serve a subset of users, in order to remove a potential bottleneck and reduce the communications overhead and cost incurred by one central node.
- o The Archive Nodes, are front-end or interface computers of the data archives, hosting the LQS functions.

As indicated by the figure the pilot configuration provides access to the IUE, EXOSAT, HST, GDF and Simbad archives as was the original aim. The

ESA-IRS database have been replaced by the ESISBIB database and furthermore the ESIS Cats and Logs database has been added. Both these databases are located at ESRIN but may conceptually be seen as individual archives, each with their own dedicated server (Local Query Server).

The ESIS Cats & Logs (ESISCAT) database contains a partially homogenized collection of the astronomical catalogues and observation logs contained also in IUE, EXOSAT, and STARCAT/HST, plus a number of new databases developed within the ESIS project. ESISCAT was introduced into the originally fully distributed architecture in order to overcome various identified problems:

- o In order to enable automatic join operations involving catalogues maintained in different archives, large amounts of data (potentially complete catalogues) may have to be transferred between archives using the telecommunications infrastructure.
- o To allow join operations between different catalogues these must be homogenized at least wrt. the join attributes, a task which if carried out dynamically can be very CPU demanding.
- o Furthermore it is questionable whether communication reliability is sufficient for frequent transfer of very large data sets, and finally the communication cost can not be neglected.

All of the above problems could effectively result in unacceptable response times rendering the system useless. By creating a centralized database of the archive directories (the logs and catalogues) and homogenizing a limited number of key attributes, these problems could be avoided. It does raise questions about the validity of the original fully distributed architecture, and the extent to which the pilot project confirms it.

The above problems can ideally be solved by having a dedicated communications infrastructure providing high-speed data transfer and making use of known distributed database techniques, e.g. semi-joins and semi-outer-joins, to limit the amount of data which is actually transferred between the individual archive nodes. However this will also require a very intelligent query optimizer, so at present it seems realistic to conclude that a certain directory information must be available in a few central access points, while the actual data can be maintained under the control of the archive in-

stitutions.

The ESIS Bibliographic database (ESISBIB) has been created to host bibliographic data dedicated to the two disciplines in the current scope of the QE. Rather than a traditional bibliographic DBMS, it has been implemented using a full text retrieval system, and as such serves the purpose of evaluating this approach against the more traditional DBMS'.

3.2 The User Platform

The User Platform provides two primary functions: query construction and result handling. These are concerned with the construction and submittal of queries and the subsequent presentation and handling of results.

3.2.1 Query Construction

As mentioned previously, one of the underlying assumptions of the ESIS QE system is that it is possible to provide a common data model for a given domain, covering all the involved archives and databases of that domain. The purpose of this model is to serve as the conceptual model through which the user expresses his information needs, i.e. his queries. This idea has been pursued within ESIS along several parallel lines and before the QE project such models were established for the scientific domains.

These models, the Astronomy Query Language (AQL) and the Space Physics Query Language (SPQL), expressed as entity relationship diagrams, provide the vocabulary of discourse, i.e. the entities, attributes and relations used to model the two domains. The original AQL model may be found in [Albrecht 91], the SPQL may be found in [Giaretta 90], figure 2 depicts the upper most layer of the AQL model, as it looks today after further clarifications and refinements.

The user models are not normalized relational database model, but contain several elements requiring specific interpretation and query facilities. Among these are many-to-many relationships, implicit relations, and hierarchical attributes.

In relational databases a relational is implemented by having common attributes in the related tables (keys and foreign keys). Using SQL the user must know which attributes to use when expressing a join across a relation between two tables. With the implicit relations this is left to the ESIS QE system.

Rather than writing:

```
Select Identifier, Title, Author
From Observational_Entity, Publication
Where Publication_Code = Asc.Number and Author
      = "Smith"
```

The user may write:

Search for

Observational_Entity show Identifier

Restricted By

Referenced_In Publication show Title, Author

Which Has

Author = "Smith"

using the ESIS Conceptual Query Language (ECQL).

In this example the user need not know how the Reference_In relation is solved, and indeed it may be solved differently on different archives. Furthermore the relation may not be solvable on a single archive but only across two or more archives. In formulating a given query using the domain model the system pilot system currently provides two means of interaction:

Query By Command,

i.e. where the user types a query using the ECQL and the entities, attributes and relations of the discipline specific domain model. In : **Search for Publication show Title, Author Restricted By Written By Scientist which has Name = 'Jones' show Address**, the underlined terms denote entities, attributes and relations in the domain model.

Query By Menu,

the entity relationship diagram is represented in a set of menus. By selecting entities and relationships from these menus it is possible for the user to construct a valid query without having to know the precise format of the query language.

Figure 3 depicts part of the current ESIS QE user interface, where both the Query By Command and the Query By Menu windows are open.

It is furthermore planned that the ESIS QE will support:

Query By Diagram,

which is directly related to the graphical representation of the entity relationship diagram. From a graphical representation the user uses a mouse to select entities

and relationships from the diagram, thereby formulating a skeleton query which is refined by adding conditions and projection clauses[GESI].

Query By Form,

provides a set of predefined query skeletons expressed as a set of forms which is refined by adding selection clauses for the involved attributes. In terms of the datamodel the Query By Form means of interaction provides a set of views. In the pilot ESIS QE system this means of interaction will be the least emphasized, consisting mainly of a set of predefined query templates.

3.2.2 Special Functions

In support of the discipline specific requirements the user may apply a number of special functions. In the Astronomy domain the core ones are concerned with the correlation of star coordinates, enabling the application fuzzy joins between star or observational catalogues. Example functions are:

In_Cone (Coord_1, Coord_2, Radius)

This function take as input the coordinates of two observations and a radius, and returns true of Coord_2 is within the cone defined by Coord_1 and the radius. The function may be used to search in individual catalogues or to join two catalogues by coordinates.

In_Box and Nearest_Object allow similar functionality to be expressed.

In Space Physics similar functions are available to compare and join observation series ordered by time, in order to investigate time and location specific events across missions.

3.2.3 Result Handling

Once the user has completed a query, it is submitted for execution. This is handled by the Query Executor, which will formulate the required subqueries, issue them to the Local Query Servers, and when a result is available inform the user.

Query results will either be table oriented or consist of data and descriptor files for images, spectra, raw data and the like. These results are initially put into the Personal Database of the user, hosted at the Access Point. Using the result handling facilities, the user may select any result and decide whether and how to have it presented on his User Machine, or

decide to transfer it to e.g. the Correlation Environment. At present the system supports table presentation and very basic image presentation facilities, implemented using public domain packages. Figure 4 shows the Table Result Window.

3.3 The Access Point

The Access Point hosts the Query Executer and Result Management components. The primary task is to decompose user queries, submit them to the archive nodes, collect and integrate the results, and manage the results belong to various users.

The decomposition process can be described in the following steps:

1. The query is analyzed, and implicit join conditions are made explicit and many to many relations are solved using a normalized conceptual data model.
2. The query is converted into an extension query, i.e. the mapping from entities and attributes of the query is used in order to find the set of extension objects (i.e. archives, entities and attributes) which contain information relevant to the query components.
3. The minimal set of archives necessary and sufficient to solve the query is identified.
4. The preferred set of archives for solving the query is selected.
5. The query is decomposed into sub-queries. A sub-query either address one specific archive and is expressed on the logical data model for that archive, or the sub-query combines the results from the previous sub-queries. A present only very basic optimization is done in order to minimize data transfer between archives and the access point.
6. Each sub-query is sent to the appropriate archive local query server where the final mapping to the physical model and DBMS is made, and a host query is issued.
7. The result from the archive is collected by the local query servers, converted to a common format and transferred to the Access Point, where further script commands or sub-queries in the standard algorithm may combine results or operate on these results.
8. The final result is obtained, stored in the users Personal Database on the Access Point, and a notification indicating the size of the result is returned to the User Platform.

In addition to the modules directly related to the primary functionality of the ESIS QE, a major part of the software in the access point is required to manage the client server interprocess communications, the user and result management and general low level process handling.

The interprocess communication is based on DECnet and TCP/IP. On top of these a dedicated communications layer has been implemented (the Application Communication System) which provides a common Application Programming Interface (API) across the various platforms, i.e. SUN/UNIX and VAX/VMS, and a gateway allowing interprocess communication between a SUN/UNIX user platform running the TCP/IP protocol and VAX/VMS DECnet based environment.

3.4 The Archive Nodes

The physical access to an archive is performed by a dedicated Local Query Server which solves the query in the following steps:

1. The query received from the query executer is mapped on to the local schema.
2. Special functions are analyzed to see whether they are supported by the archive. Currently only functions supported by the archive is handled but with some limitations the special scientific functions could be applied in the LQS external to the archive.
3. A connection to the archive is established and a series of commands representing the query is issued.
4. The result is received and converted into the format of the common RDBMS, currently ORACLE, before it is returned to the access point.

Seen from the Query Executer each archive node is in principle a relational database with standard RDBMS capabilities, i.e. search and join functions. This implies that result combination involving temporary results can take place at any archive node. However, due to the very diverse nature of the host DBMS, ranging from commercial RDBMS, over archive

specific DBMS's to the ESISBIB full text DBMS, this can not be fully achieved within the current project.

4. Current Status and Conclusion

The coding of the ESIS QE started in January 1992, and at the current moment (August 1992) a first development release is forthcoming.

As the system becomes more stable it is the intention to release it, initially to a small set of select users, prior to the public evaluation and distribution.

The evaluation of the Pilot Project will take place in the first and second quarter of 1993, at which time the development of the ESIS QE pilot project will be completed.

5. Acknowledgements

The authors wish to thank everyone within the ESIS team at ESRIN, within the industrial Consortium and at the various scientific institutions involved in the ESIS project. They have all in one way or another contributed to this paper and the results achieved thus far..

6. References

- [Albrecht]
Miguel A. Albrecht
ESIS A science Information System
in "Databases & On-Line Data in Astronomy"
ed. M. Albrecht and D. Egret.
Kluwer Academic Press, 1991.
- [Giaretta]
D.L. Giaretta et al
Rutherford Appleton Laboratories and
University of Sheffield
ESIS Draft Final Report The Space Physics Query
Language
- [GESI]
GESI
The KIM Conceptual Query Language
GE-05-06-008
July 16, 1990

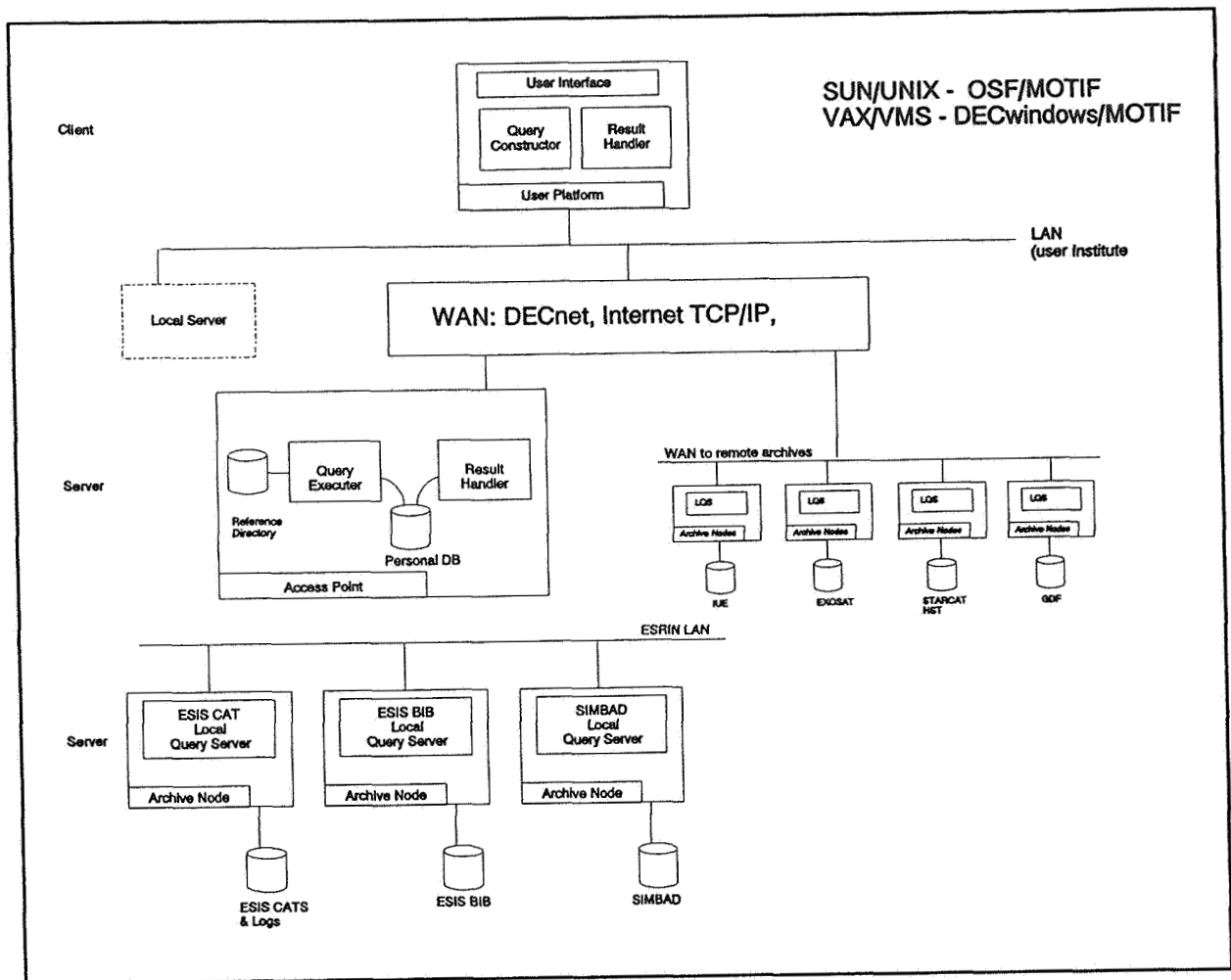


Figure 1: The ESIS QE Infrastructure

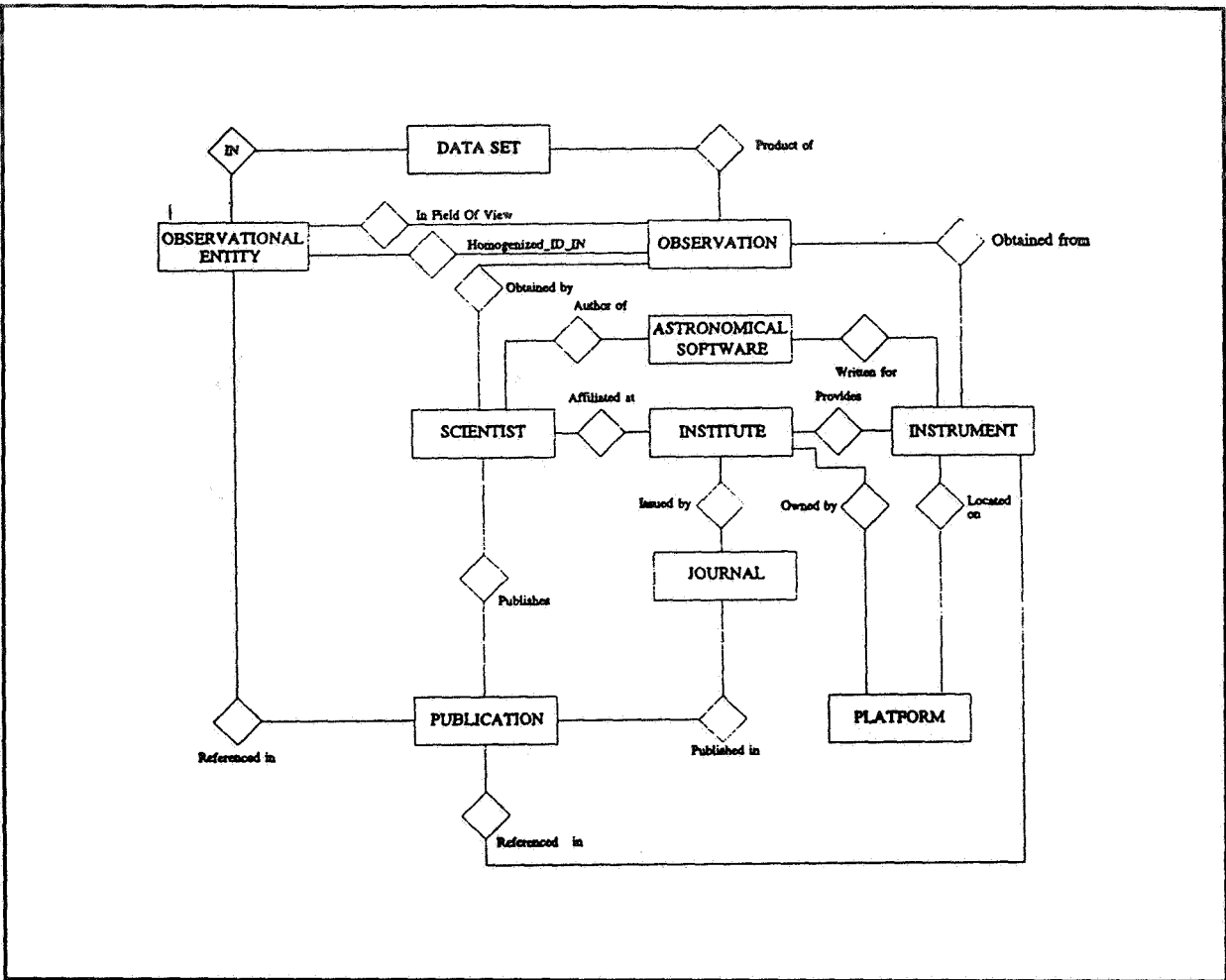


Figure 2: The AQL Datamodel

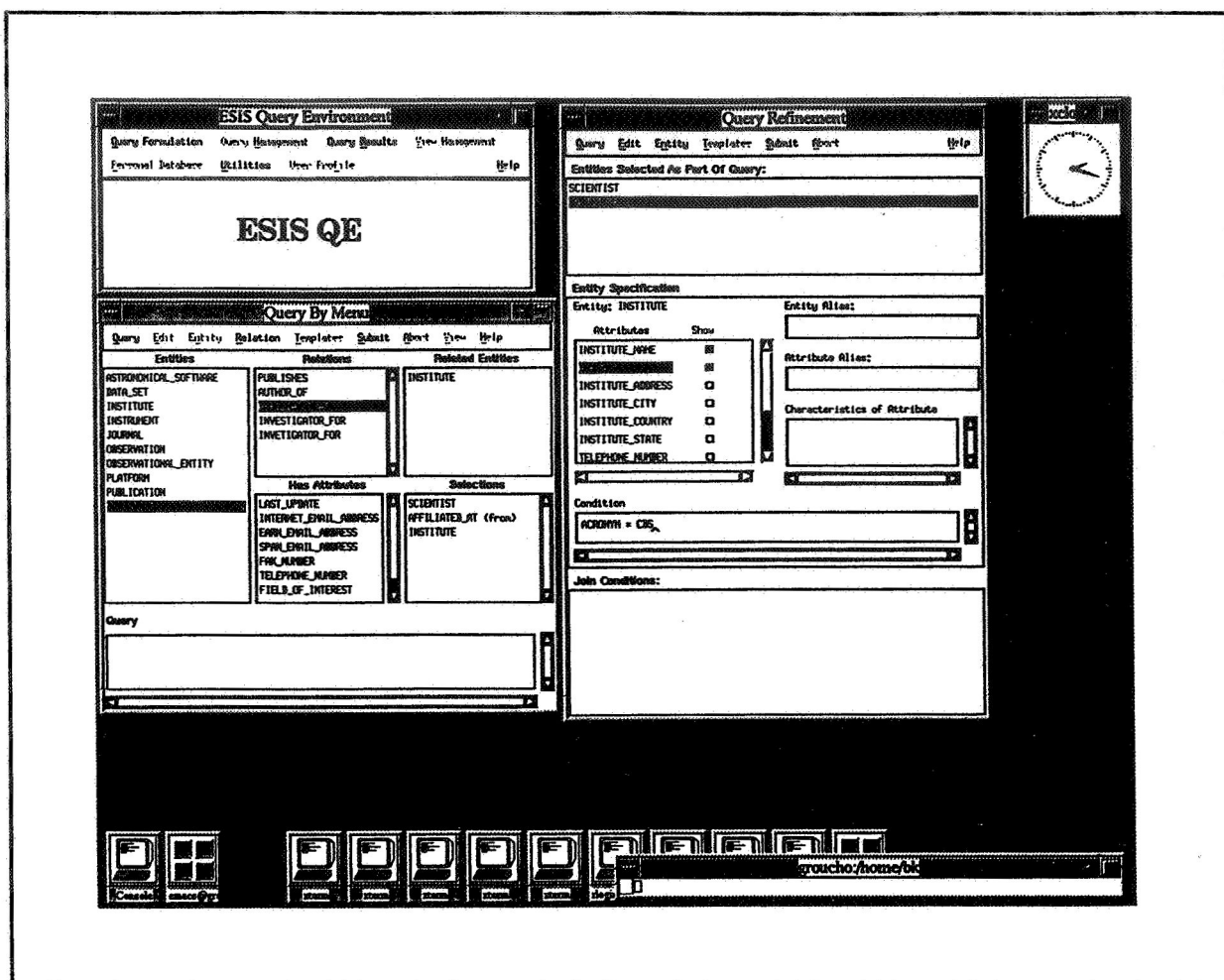


Figure 3: The ESIS QE Main Menu, Query By Command and Query By Menu Windows

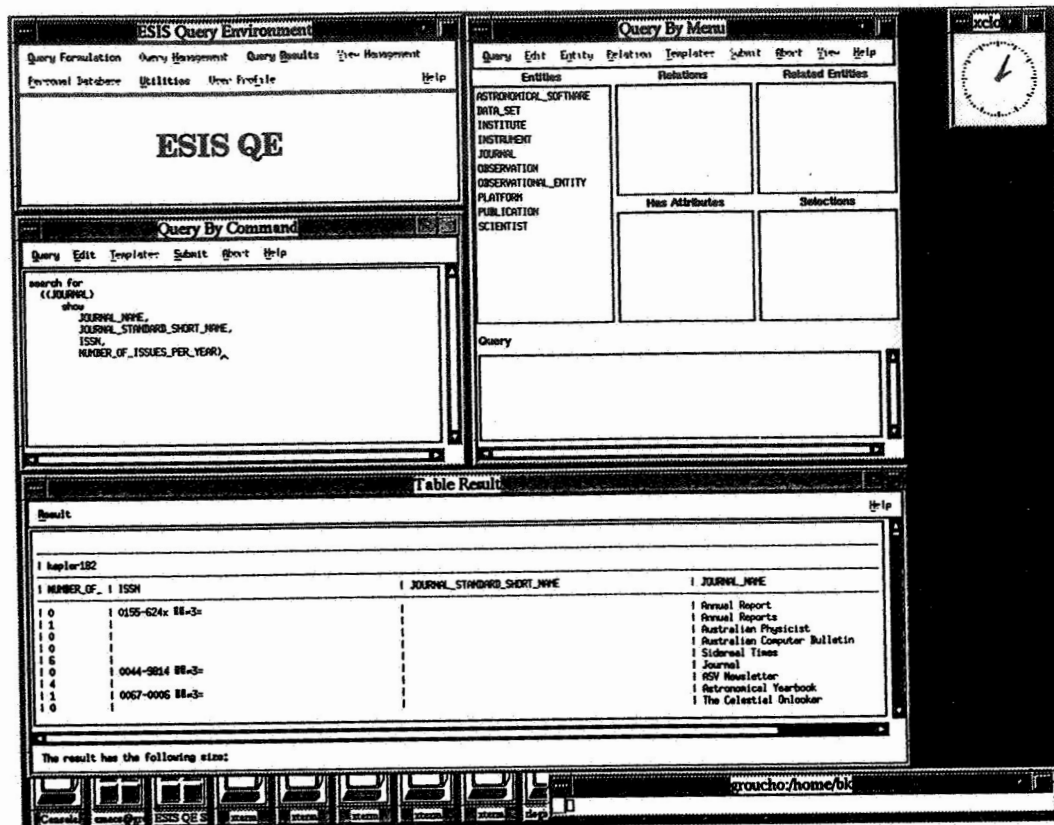


Figure 4: The ESIS QE Table Result Window

MARS OBSERVER DATA PRODUCTION, TRANSFER, AND ARCHIVAL: THE DATA PRODUCTION ASSEMBLY LINE

David B. Childs

N 9 4 - 2 3 9 5 3

Jet Propulsion Laboratory
Pasadena, California, U.S.A.

ABSTRACT

This paper describes the data production, transfer, and archival process designed for the Mars Observer Flight Project. It addresses the developmental and operational aspects of the archive collection production process. The developmental aspects cover the design and packaging of data products for archival and distribution to the planetary community. Also discussed is the design and development of a data transfer and volume production process capable of handling the large throughput and complexity of the Mars Observer data products. The operational aspects cover the main functions of the process: creating data and engineering products, collecting the data products and ancillary products in a central repository, producing archive volumes, validating volumes, archiving, and distributing the data to the planetary community.

1. INTRODUCTION

1.1 Purpose

The purpose of this paper is to describe a proposed archive collection production process for the Mars Observer Mission. The Mars Observer archive collections are logical assemblages of science data products and their associated meta data, ancillary data, documentation, catalog information, and software that were identified for permanent archive and distribution to the general planetary science community. The contents of the archive collections were agreed upon by

Mars Observer and the Planetary Data System [Ref. 1-2]. The Planetary Data system acts as the contact between Mars Observer and the National Space Science Data System which serves as the permanent archive for planetary science data in NASA. Mars Observer is the first mission to be fully integrated with an operational Planetary Data System.

The archive collection production process includes the extraction of raw and reduced data products and their associated ancillary products from a centralized Project Database, repackaging the data products in some cases, organizing the products into archive volumes, premastering CD-ROMs, validating archive volumes, and coordinating the mastering and release of CD-ROM volumes. The proposed functional design of the archive collection production process was prepared by the Science Integrated Data System Task at the Jet Propulsion Laboratory (JPL). This team is a multiorganizational design team with representatives from the Mars Observer Project, the JPL Multi-mission Operations Systems Office, the Planetary Data System, and the JPL Standards Group. The functional design will be presented to the four organizations in December, 1992. Prototyping and detailed design began in October, 1992 by the Mars Observer Data Administration Team and the Science Data Validation Team.

1.2 Summary of the Mars Observer Mission

The Mars Observer Spacecraft was launched

in September of 1992. It will be placed into orbit around Mars in August, 1993. After a four month orbit insertion phase, the instruments on the spacecraft will begin mapping the atmosphere, surface, and interior for at least one martian year (687 Earth days). The spacecraft has seven instruments: the Gamma Ray Spectrometer, the Magnetometer/Electron Reflectometer, the Mars Observer Camera, the Mars Observer Laser Altimeter, the Pressure Modulator Infrared Radiometer, the Thermal Emission Spectrometer, and Radio Science. The objective is to produce a suite of data products that depict atmospheric, surface, and subsurface characteristics as a function of latitude, longitude, altitude, and season.

The Mars Observer ground data system includes a centralized mission operations component at JPL and a distributed component located at the home institutions of the Principal Investigators, Team Leaders, Interdisciplinary Scientists, Team Members, Co-Investigators, and Participating Scientists. Science Operations Planning Computer (SOPC) workstations will be electronically connected via NASCOM links to a centralized Project Database Personnel at the central site and those with remote workstations will use the Project Database to coordinate observation planning, deposition of raw science and engineering data, acquisition of data, deposition of reduced data and documentation, and extraction of reduced data for analysis.

The collective science teams and mission operations teams on Mars Observer will produce approximately 80 unique data products totalling to about 370 GB of data. The data and ancillary products will be organized into 15 archive collections: one for each of the six instruments and the Radio Science Experiment, one for each of the six Interdisciplinary Scientists, one for the

SPICE data (geometry data), and one for the engineering data. The archive collections will be stored on approximately 800 CD-ROM volumes.

The overall archive collection flow production is shown in Figure 1. Mars Observer is responsible for the production and validation of the archive collections and archive volumes (CD-ROMs). After a generation/validation period of six months (since the creation of the data product) the CD-ROMs are released to the Planetary Data System for distribution to the NASA sponsored planetary community and permanent archival. The National Space Science Data Center serves as the permanent archive and also distributes to the general science community. The archive collection production process includes only the tail end of downlink portion of the Mars Observer (MO) mission operations flow. The included functions are shown in Figure 2.

2. WHAT IS AN ARCHIVE COLLECTION

An archive collection is a logical assemblage of related data products and all the information necessary to access, understand, and use the data (fifteen archive collections for MO). Archive collections are organized into data sets (approximately 80 for MO) and data sets are groups of data product instances. Each data product contains the actual values for the measured parameters for one observation (for example, an image) and its associated meta data. The meta data identifies the observation, defines its properties, and defines the structure and format of the data values. Data sets may include unique versions of geometry data or calibration data. A data set may also include documentation catalog information, and software.

Documentation may include flight project documents, instrument papers, science

articles, or any other textual material deemed necessary to understand and use the data sets or software. Catalog information consists of completed, predefined catalog loading templates or outlines for describing missions, spacecraft, instruments, data sets, and personnel. Software includes executables, source code, installation procedures, user guides, and design documents. SPICE Kernels (or geometry data) consists of software and data files that together characterize the critical features of the Spacecraft ephemeris, Planet ephemeris or other target, Instrument characteristics, C-matrix/scan platform orientation, and Events associated with data sets. Calibration data is any data used in the calibration of the instrument which is necessary to understand or use the data sets or software. Finally, in some cases, product specifications such as Science Data Product Software Interface Specifications (SISs) or Archive Collection SISs may be included.

The data sets are packaged onto physical volumes such as CD-ROM, CD-WO (write once), or magnetic tape for archival and distribution. Most data sets require several volumes. Volumes are grouped into volume sets. Each volume and volume set has a unique identifier. Each volume contains one of several standardized directory structures, required files such as AAREADME.TXT and one containing a volume description, and a tabular index of the data products on the volume. Multivolume sets contain cumulative indices. The organization of data sets and volumes is standardized and described in the PDS Data Preparation Workbook, Volumes I and II [Ref. 3-4].

The organization and expression of the meta data for the data objects is standardized. The meta data identifies the observation, describes its characteristics, associates it with other objects, and gives the internal

organization and format of the data values. The meta data is contained in a catalog object. It is expressed in a "keyword equals value" notation (for example, TARGET = MARS) called the Object Definition Language [PDS 92]. The keywords and standard values are defined in the Planetary Science Data Dictionary [Ref. 5]. The keywords describing the internal organization of data objects are aggregated into structure objects (such as IMAGE, TABLE, QUBE). A wide variety of structure objects are standardized for use in the meta data for primary and secondary data objects.

3. OBJECTIVES OF THE MO ARCHIVE COLLECTION PRODUCTION PROCESS

The Mars Observer 'archive collection production system has the following objectives.

- (1.) Produce archive volumes containing quality data with the meta data and ancillary products necessary to access, understand, and use data. The volumes must be on a permanent, high density, and distributable media. The products and volumes must adhere to Planetary Data System standards.
- (2.) Produce all the archive volumes before the end of the mission (or the accounts close). Restoration is very expensive.
- (3.) Be cost effective, automated, and reliable. Development and operations must be accomplished within a fixed budget. The process must be automated where possible to accommodate the high throughput for Assemble Volumes with some products and validate Volumes for all volumes.

4. MARS OBSERVER ARCHIVE COLLECTION PRODUCTION PROCESS

The high level functions of the MO archive collection production process are shown in Figure 3. These functions span from the creation of the bested telemetry packets through to use of the archive volumes by the general planetary community. Several of the functions are existing subsystems in the MO Ground Data System and are provided by the Multimission Operations Systems Office (MOSO). The archive production process was designed to minimize the impact on these existing functions. The archival and distribution function is performed by the operational Planetary Data System. The use of archive volumes is performed by the general planetary community. While these functions are not a part of the proposed functional design they are included in the description of the archive process because they are the interfaces to the proposed functions, the proposed functions require minor modifications to the existing functions, and they are required to understand the overall archive process flow.

The remaining functions, the package, assemble, validate, and coordinate release functions were designed recently for completion of the archive collection production process and to fully integrate the mission data output with the archive system input. The following sections describe the high-level archive collection production process (See Figure 3).

4.1 Collect Telemetry and Create Level 0 Engineering and Science Products

Telemetry packets from the Deep Space Network arrive at the Collect Telemetry and Create Level 0 Engineering and Science Products function. This function is an abstraction of the existing front end of the

downlink of the ground data system. This function performs Reed/Solomon decoding, frame synchronization, extracting packets from frames, prepares channelized engineering data, performs besting of the data, and extracts anomaly records. The telemetry packets are stored on the centralized Project Database. The output of this function is the final or bested science and engineering telemetry packets (Level 0 products). This function is performed by the Mission Operations System Teams.

4.2 Create Science Data and Ancillary Products

The telemetry packets are extracted from the Project database by the Principal Investigator Teams and transferred to their remote sites in the Create Science Data and Ancillary Products function. Then the data is decompressed, reconstructed, and reformatted. Various error corrections, calibrations, and science processing algorithms are applied to produce reduced science data products (Level 1+). This function is an abstraction of the science investigations performed by the Principal Investigator and Interdisciplinary Science Teams. The associated meta data is packaged with the data products and it is wrapped with Standard Formatted Data Unit (SFDU) labels. The data product is then ready to be transferred to the Project Database.

4.3 Package Level 0 Engineering and Science Data Products

The Package Level 0 Engineering and Science Data Products systematically extracts the final level 0 telemetry (science and engineering) from the Project Database as the database nears its capacity and packages it for off-line storage and archival. Since the telemetry packets are stored in a DBMS, they must be extracted with queries based on a

predetermined extraction strategy (e.g. one orbit's worth or one day's worth). Each group of packets is written to a file, the associated meta data for the group is included, and the product is packaged into an SFDU. Other ancillary products and archive volume components such as documentation and volume indices for the Level 0 archive volumes are prepared here. The output of this function is a logical archive volume containing level 0 engineering and science data products ready for premastering onto CD-WO. This function is one of the new, proposed functions and would be performed by the MO Data Administration Team.

4.4 Collect Data and Ancillary Products

The Collect Data and Ancillary Products function receives completed, reduced, science data products from the science teams, performs a cursory check of the packaging and meta data, catalogs the files based on values in the meta data, and stores them in a file management system (i.e. loads them into the Project Database). This function also accepts queries for the retrieval of data products. The Project Database is managed and operated by the MO Data Administration Team.

4.5 Assemble Volumes

The Assemble Volumes function creates a manifest for each archive volume based on the volume specifications in the Archive Collection Software Interface Specifications (SIS). The manifests are expressed in keyword equals value notation and are human and machine readable. Using the manifest, the specified data products and ancillary products are extracted from the Project Database. The extraction step is verified and the manifest updated (some desired products may not be ready). Several validation checks are performed to ensure

that the products conform to standards. Some products may need to be reformatted because of chronic errors or known design compromises. Reformatting is performed based on instructions in the manifest. Some ancillary products may need to be repackaged because they were stored on the Project Database as a single aggregate unit (e.g. software source code). Then, the volume specific components are completed such as the AAREADME.TXT file, the volume description, and the volume indices. Next, a logical volume is created and used to premaster a CD-ROM volume image. The volume image is written to CD-WO. The capability to write the image to 8mm tape is also provided. Finally, the media is checked to verify the premastering steps. The level 0 engineering and science products enter this function as a completed logical volume and only go through the premastering and verification steps. This is one of the new, proposed functions and would be performed by the MO Data Administration Team.

4.6 Validate Volumes

All of the completed archive volumes are sent to the Validate Volumes function for final validation before release. In this function, the file and subdirectory contents of the volumes are compared with the manifest and the volume index. The meta data in the data products are compared to the design of the data products in the Science Data Product SIS and the Planetary Science Data Dictionary [Ref. 5]. The volume organizations are compared to the designs of the volumes in the Archive Collection SISs. This function does NOT systematically check the quality of the science data. However, statistical sampling may be used to verify the correlation of the meta data with the data products. Also, a final error analysis is performed using the cumulative error log which accompanies each volume. The error logs are expressed in

keyword equals value notation and are human and machine readable. The results of the error analysis determine if the volume is released or returned for rework. A validation report is prepared. The error analysis also determines where in the process to return the volume for rework. This function may also perform minor error corrections and premastering of volumes. This is one of the new, proposed functions and would be performed by the Science Data Validation Team.

4.7 Coordinate Release of Volumes

The Coordinate Release of Volumes function receives all of the completed, validated archive volumes and their validation reports. In this function, the completed volume and the validation report are reviewed by the MO Data Archive Working Group to determine if the volume should be released, checking that the six month generation/validation period has passed (MO policy), and obtaining approval from the Principal Investigator Team representative. If the volume is not ready for release, it is held for rework or pending further negotiation. When the volume is ready for release it is either duplicated in the Assemble Volumes function or sent to a CD-ROM mastering vendor. If less than fifteen copies are required, it is more cost effective to produce CD-WO copies than to master and duplicate. If mastering is required, the CD-ROM artwork, packaging artwork, distribution lists, and the purchase order are prepared for the mastering vendor. A copy of the mastered volume may be requested for verification before duplication. Upon completion of the mastering and duplication, any copies received by MO are distributed internally. The Planetary Data System and the National Space Science Data Center receive fifty to several hundred copies directly from the mastering vendor. The CD-WO copies produced internally are packaged

manually and the appropriate number of copies are distributed to the Planetary Data System directly. This is one of the new, proposed functions would be coordinated by the Science Data Validation Team.

4.8 Archive and Distribute Volume

The Archive and Distribute Volumes functions are performed by the Planetary Data System (PDS) and the National Space Science Data Center (NSSDC). The PDS and the NSSDC receive copies of the final archive volumes either from the mastering vendor for large duplication orders or from MO directly for limited copies of CD-WO volumes. The PDS operates interactive catalogs of the archive volumes and the contained data sets. In some cases, the PDS has catalogs which inventory the archive volumes down to the individual data product level. The PDS accepts orders and distributes to the NASA funded planetary science community. Copies of the volumes are archived in the PDS local archives at its Discipline Nodes. The PDS scientists representing the six planetary science disciplines and the navigation area provide expert help in finding and using archive collections. Special processing of data sets is provided when requested. The PDS maintains standards for the contents and organization of archive collections [PDS 92] and provides (archive) data engineering support to missions. The NSSDC serves as the permanent archive for the planetary science archive collections. It also accepts orders and distributes to the remainder of the NASA funded scientists and the general public.

4.9 Use Archive Volumes

The general planetary science community requests data sets, data products, expert help, and special processing to support scientific

investigations. The community also provides feedback on the accessibility, usability, and quality of the archive collections. Over time, the feedback is formalized into refinements to standards for preparing archive collections.

5. CONCLUSIONS

The development of the MO archive collection production process is currently in the detailed design phase for the package level 0 products, assemble volumes, validate volumes, and coordinate release functions. The remaining functions in the described archive process already exist. One of the key development objectives is to minimize the impact on the existing functions. The complete functional specification of the four proposed functions is contained in the "Mars Observer Archive Collection Production Specification" document [Ref. 6] prepared by the Science Integrated Data System Team. The functional specification will be reviewed by Mars Observer, the Multimission Operations Systems Office, and the Planetary Data System in December 1992. The Principal Investigator Teams and the Data Archive Working Group are in the process of designing the data products and CD-ROM volumes. Prototypes of the level 0 engineering archive volumes (CD-WO) were produced recently using the Package Level 0 Engineering and Science Data Products function and portions of the Assemble Volumes function.

The development process has provided a testbed for several archive production "technologies". These are:

- (1.) The approach for writing Archive Collection SISs which contain the contents and organization specifications for over 800 unique CD-ROM or CD-WO volumes.

- (2.) The means of expressing the volume specifications in machine and human readable form as manifests. The manifests will support automated extraction of products from the Project Database, automated volume assembly, and automated volume validation.
- (3.) The approach for describing and tracking errors in a machine and human readable form throughout the production process.
- (4.) The approach for scheduling and tracking volume assembly, volume validation, and release throughout the production process.
- (5.) An approach for integrating several disjoint data production) and validation tool sets into an efficient, automated assembly and validation process.

The Science Integrated Data System Team proved to be very successful as a multiorganizational system engineering and design team. Engineers representing Mars Observer, the Multimission Operations Systems Office, the Planetary Data System, and the JPL Standards Group worked together effectively to define the archive collection production process and bridge the gap between the MO ground data system and the Planetary Data System. Implementing the proposed archive production functions will ensure that the Mars Observer science and engineering products will be archived and distributed to the general planetary community in a timely manner. In addition, the enormous costs of data restoration can be avoided for the 370 GB of MO data to be archived.

6. REFERENCES

1. Mars Observer Project Data Management Plan, 642-33, October 1991, JPL D-6615.
3. Planetary Data System Data Preparation Workbook, Volume 1, Procedures, November 1992, JPL D-7669.
2. Mars Observer Archive Policy and Data Transfer Plan, 642-447, March 26, 1992.
4. Planetary Data System Data Preparation Workbook, Volume 2, Standards, November 1992, JPL 3-7669.
5. Planetary Science Data Dictionary Document, November 1992, PDS Version 2.0, Rev. B, JPL D-7116.
6. Mars Observer Archive Collection Production Specification, December 1992.

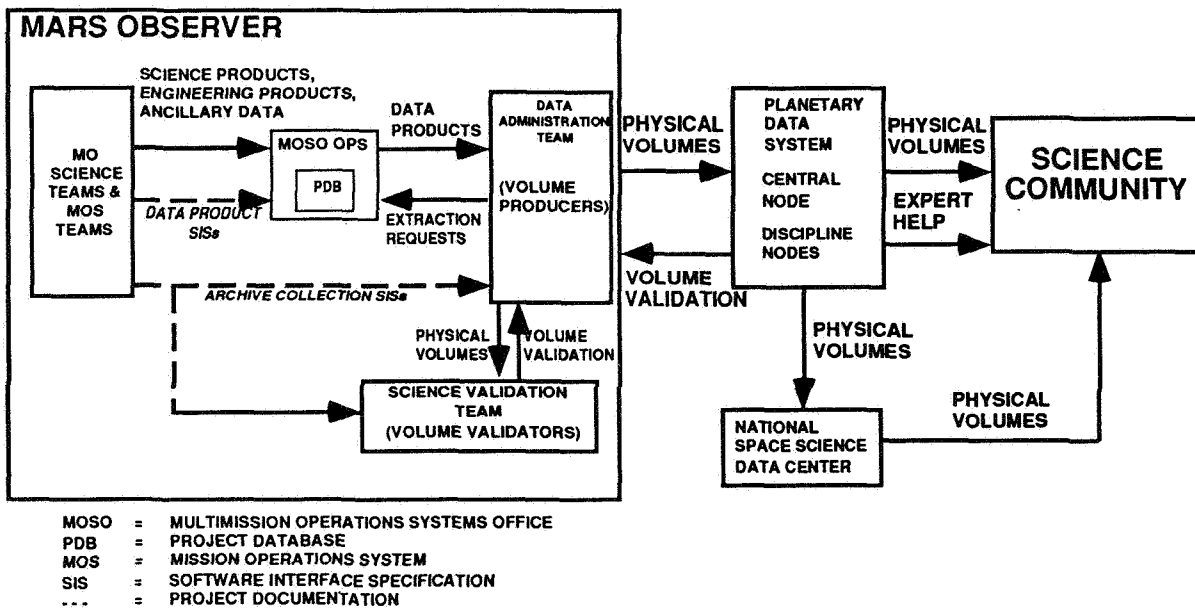
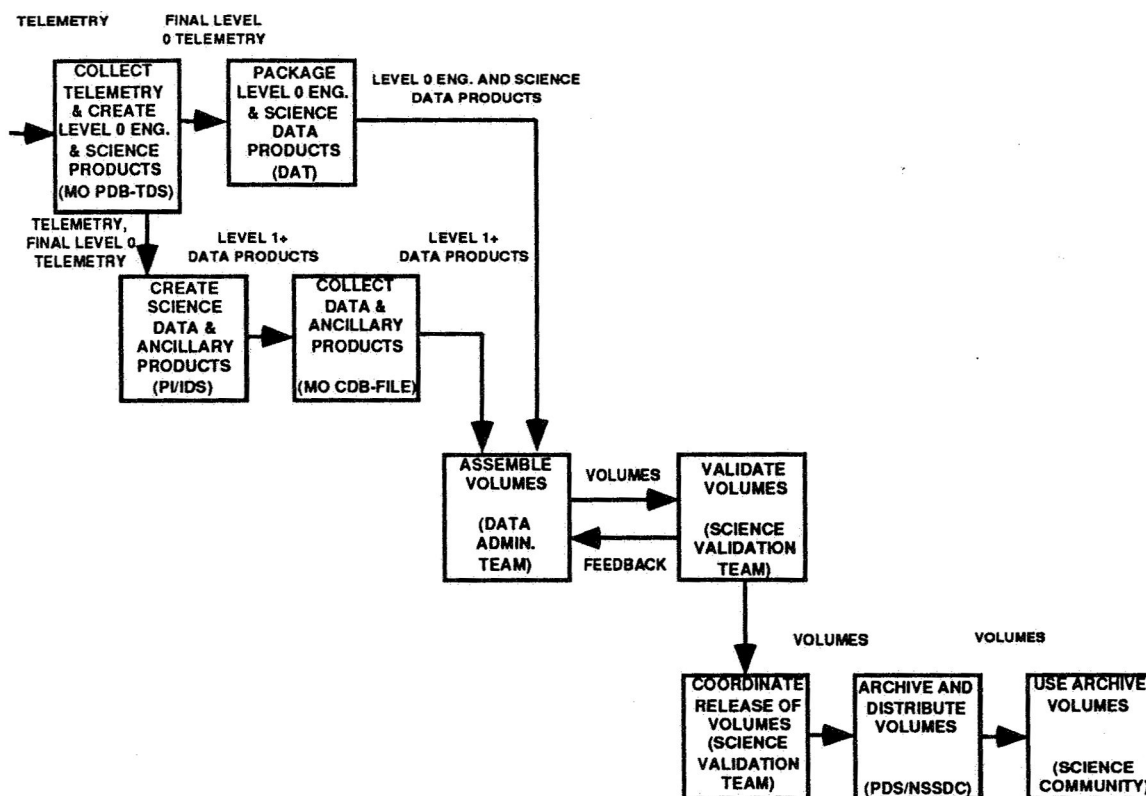
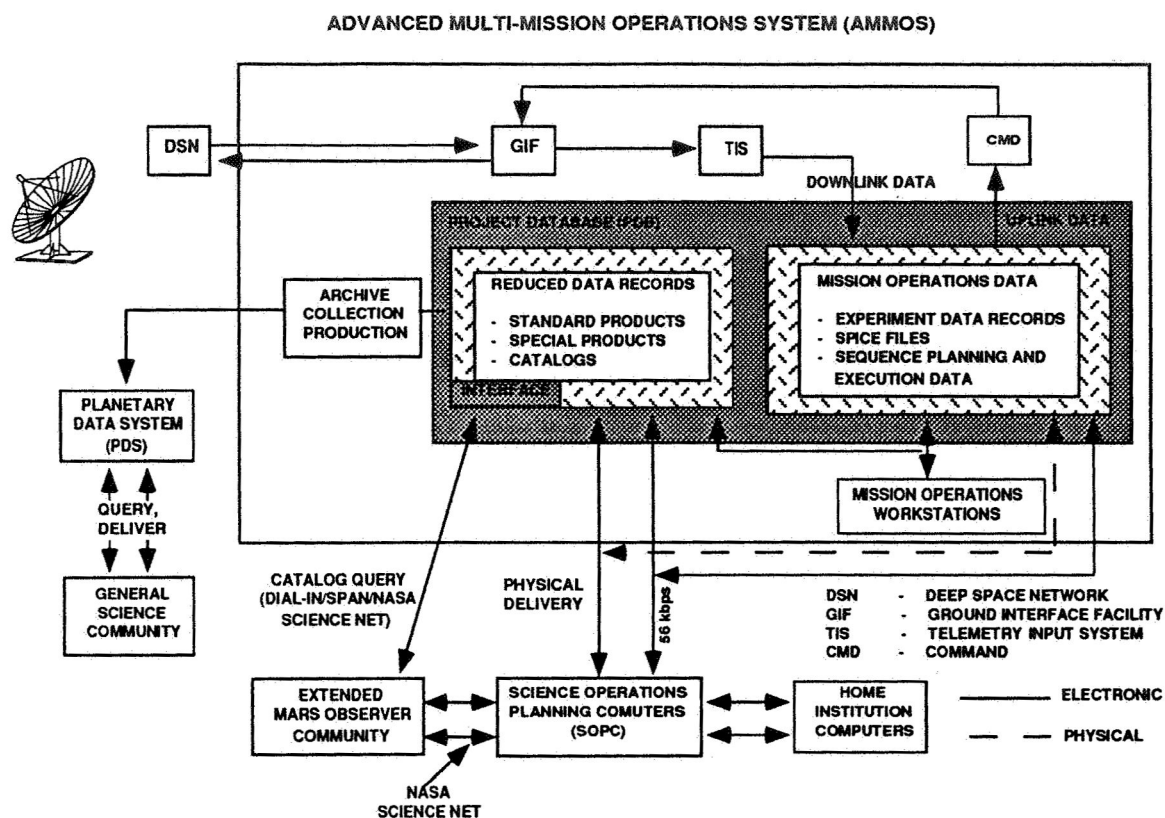


Figure 1. MO Archive Collection and Volume Flow



THE EUROPEAN SPACE AGENCY
STANDARD FOR
SPACE PACKET UTILISATION

N 94 - 23954

J.-F. Kaufeler¹, A. Parkes², A. Pidgeon³

¹ ESA/ESOC/FCSD, 6100 Darmstadt, RFA.

² NOVA Space Ltd., Bath, UK

³ VEGA Space Systems Ltd., Harpenden, UK

ABSTRACT

This paper presents the ESA concept for the use of CCSDS defined Telemetry and Telecommand Packets at the application level. These Packets are used to monitor and control remotely a spaceborn application. This concept is defined in a Packet Utilisation Standard (PUS) which should become applicable for all ESA missions using Packets. The production of this standard is under the responsibility of an ESA standardisation group called "COES".

Keywords: Packet Utilisation, Packet Type, Parameter, Packet Structure, COES.

1. INTRODUCTION

The usual way to monitor a spacecraft, is to get parameters via the Time Division Multiplexed Telemetry. Similarly the control function is performed by telecommand frames containing basic instructions to the spacecraft. This constitutes actually a remote monitoring and control of on-board electronics.

However, on-board software is used more and more to address specialised complex functions which could otherwise hardly be operated from ground.

Up to now, because of no other choice, on-board software is remotely monitored and controlled using the classical electronic oriented sampling mechanism and bus commands. Rapidly it became clear that this imposes heavy constraints on the on-board software implementation, limiting its flexibility and consequently hampers the trend towards more on-board intelligence and autonomy.

In order to overcome this situation, the Consultative Committee for Space Data Systems (CCSDS) recommended the

use of Telemetry (TLM) and Telecommand (TLC) Packets and specified precisely how such Packets should be transmitted between space and ground. Those Packets will mostly contain software data of which structure and format will have a significant impact on the software architecture on board, but also on ground.

However, making things more flexible leads to the possibility of implementing a given control function in many different ways. Anticipating this potential problem, ESA decided to address this point in one standardisation committee (COES) and to produce a Packet Utilisation Standard which recommends to users of Packets general rules governing the exchange of Packets between space and ground, as well as the data structure and format to be used within the Packets. It also recommends specific Packets for specific but commonly used functions.

This standard should also enable the implementation of a clean software architecture on board and on ground much more adapted to on-board intelligence and autonomy.

2. PROBLEM CAUSED BY THE USE OF PACKETS

Until a few years ago, satellites had only limited on-board processing capabilities. The Telemetry was essential on-board device digital read-outs (e.g. sensors, switches, payload hardware etc.) sampled at fixed regular intervals and multiplexed at fixed positions in a telemetry frame. Similarly the Telecommand contained fixed length data words which were supposed to load on-board registers (for a specific device) or to enable/disable switches. With this technique it was possible from ground to monitor and control remotely on-board devices (Fig. 1). However, the as-

sociated space-ground communication technique of the Telemetry and Telecommand data did not guaranty a safe and complete transmission of data.

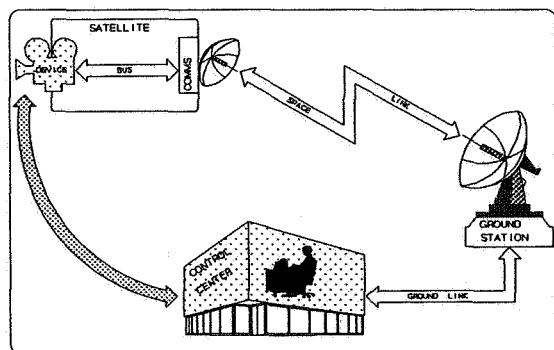


Fig.1 : Remote Control of a Device

With the advent of more complex on-board processors and thus software, the need arose to exchange data of a different nature (generated by software) between space and ground which requires a more flexible transmission capability (e.g. variable content, asynchronous generation) and consequently a transmission technique of better quality (e.g. error-free and complete). In fact, one does not control on-board devices directly but rather through on-board software. What was needed is a technique to enable a remote monitoring and control of on-board software (Fig. 2).

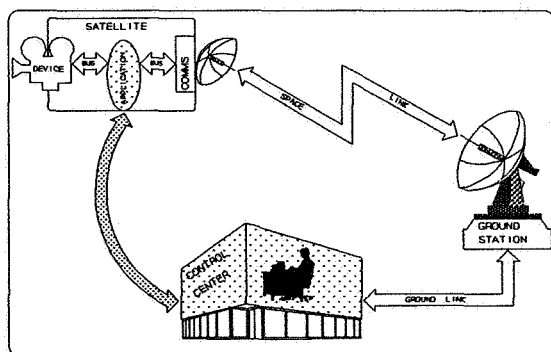


Fig. 2 : Remote Control of an Application

The Consultative Committee for Space Data Systems (CCSDS) has defined and recommended the use of Telemetry Packets (Ref. 4) and Telecommand Packets (Ref. 5) which is a space-

ground high quality communication technique enabling a flexible exchange of data between an on-board Application (usually software) and a ground system (also software). However, nothing is said on the content of these Packets and according to which principle they should be exchanged.

This is now the key problem. Since this aspect will largely determine the system architecture on board and on ground and also to some extent the functionality, it will not be possible to achieve common infrastructures (e.g. ground control centres) reusable for different missions, in absence of recommendations on the content of Packets and the way they are exchanged.

3. ROLE OF THE C.O.E.S. STANDARDISATION GROUP

In 1987 an intra-agency standardisation group was created; the Committee for Operation and EGSE standardisation (COES). The objectives were to define and specify the common functions between a satellite checkout system (EGSE) and a satellite control system. Even if these systems are used for a different objective in a different project phase, a large amount of functions are similar and the logical interface to the satellite is identical. Therefore, a common system could be used for the checkout and operation within a given project (vertical commonality) but also across projects (horizontal commonality).

It was also decided to define such a common system for missions using the newly defined Telemetry and Telecommand Packets. It became rapidly clear that this task was only feasible if a clear satellite-ground interface is specified, based on the use of Packets (Fig. 3).

Consequently, the first task of the COES was to produce a standard which defines precisely the use of Packets from an application viewpoint. Such a document (Packet Utilisation Standard STD-07-101) has been produced by the COES and is presently undergoing an Agency-wide review.

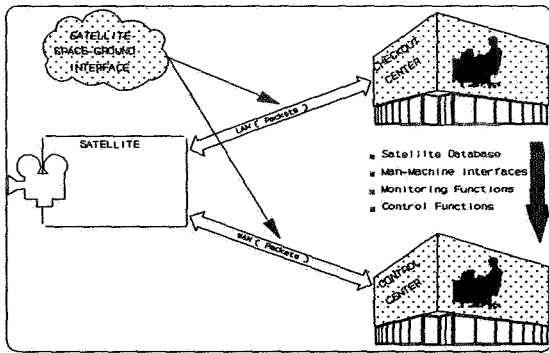


Fig. 3 : Checkout / Operation System Commonality

4. SCOPE OF THE STANDARDISATION

The Packet Utilisation Standard (Ref. 1) serves to complement and extend the Packet Telemetry Standard (Ref. 2) and the Packet Telecommand standard (Ref. 3), in order to satisfy the requirements of electrical integration and testing and flight operations. The term Utilisation is used to cover all what should be known around Packets (e.g. their content, how it is generated, which functionality they convey, etc.) in order to operate a remote Application.

An Application is an on-board logical entity which generates Telemetry Packets and also receives Telecommand Packets for the purpose of monitoring and controlling the Application. The Application is identified by an Application ID used to establish an end-to-end connection between this Application and the Ground. An Application can be any (or part of) platform of payload subsystem.

The standard is an interface document defining the relationship between space and ground. It, therefore, contains:

- requirements relating to satellite monitoring and control functions and to testability;
- Telemetry and Telecommand Packet types and sub-types (depending on the functionality);
- Packet data structures and format.

To the maximum extent, this standard utilises techniques emerging from modern software engineering trends.

This standard is in principle applicable to the full variety of mission classes, ranging from a satellite of relatively simple on-board design in an orbit with continuous ground coverage like a geostationary communication satellite, to a more complex mission like a low-earth orbiting scientific satellite or a deep-space mission.

The three principal driving factors which place the strongest demand on Telemetry and Telecommand are the degree of on-board complexity, the degree of on-board autonomy (these two factors being inter-related) and the link transmission capacity.

5. OPERATIONAL REQUIREMENTS

Telemetry and Telecommands must be provided to enable the safe and reliable execution of all nominal and contingency operations which are required to achieve the mission objectives.

The general principles of the operations model are as follows:

- Nominal operations

The Application generates the relevant housekeeping information (depending on the mission phase) either whenever a significant event occurred on-board or at regular intervals.

In the former case this information may also be generated on request by the ground. The information is composed of a suite of parameters (e.g. on-board variables, indirect device read-outs etc.).

The Application generates information reports to indicate special events and any actions taken autonomously by the on-board system.

On reception of a Telecommand the Application responds systematically on the validity status of it, and execution progress until completion.

When accurate parameter time correlation is required, this is achieved either by associating explicitly a time-tag with the parameter or by reconstructing of

the time on ground from the Packet time with implicit knowledge of the on-board "sampling" mechanism used.

- Contingency operations

The Application has a diagnostic mode which enables to investigate short timescale phenomena of transient effects.

The Application logic may be modified or reprogrammed from ground in case of malfunctioning.

The Application may also be operated in an off-line manner in order to check its correct functionality before switching it to the operational mode.

Any Application protection logic may be disabled from Ground and thus overruled.

6. PACKET TYPE AND SUBTYPE

For a given Application, the information contained in a Packet may correspond to different functionality. This is identifiable from the Packet Data Field Header (PDFH) which has a fixed structure for Telemetry and for Telecommand (Fig. 4).

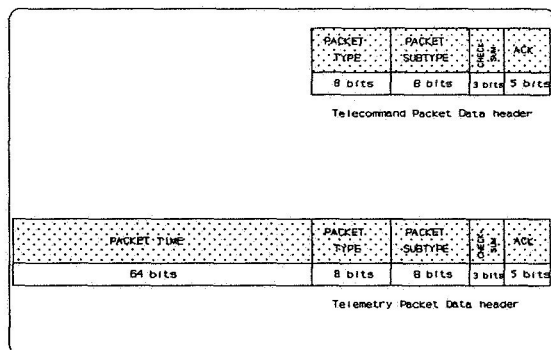


Fig. 4 : Packet Data Field Header

The two most important fields in the PDFH are the Packet Type (PT) and the Packet Subtype (PST). The PT identifies the higher level of a Packet category whereas the PST identifies a lower level within this category. The couple (PT, PST) has a unique definition and applies across different Applications.

All Packet contents (the Packet data field) are composed of fields (here called parameters) coded according to standard formats and organised according to a standard structure.

For well-defined monitoring and control functions, specialised Packets are defined, to which specific (PT, PST) pairs have been assigned, and which Packet content conforms to the general rules. Of course, those Packets are mandatory only if these functions are needed.

However, growth capability of the standard is ensured by expanding it with the introduction of new function categories or new subfunctions within an existing function category.

The presently defined categories of Packets are shown in Table 1 and Table 2.

TELECOMMAND PACKETS		
Type PT	SubType PST	FUNCTION
1	1	Device Command
2	1...5	Control Process
3	1	Software Command
4	1,2,3	Load Memory
5	1,2,3	Dump Memory
6	1,2	Diagnostic
7	1...7 11...17	Telemetry Generation
8	1...12	Master Schedule
9	1...9	On-Board Monitoring

Table 1 : List of Defined Telecommand Packets

TELEMETRY PACKETS		
Type PT	SubType PST	NATURE
1	1	Housekeeping
2	1	Report
3	1	TC Verification
4	1	Time
5	1,2,3	Memory Dump
6	1,2	Diagnostic
7		(Not defined)
8	1	Master Schedule
9	1,2,3	On-Board Monitoring

Table 2 : List of Defined Telemetry Packets

7. PACKET STRUCTURE AND FORMATS

The structures and formats adopted for this standard are oriented towards modern software language (e.g. Pascal, C, C++, ADA) data structures. Thus they can be handled conveniently by the on-board or ground software, and offer the required flexibility. The interpretation of the Packet requires an implicit knowledge of the Packet content by the application supposed to process it. This is achieved by specific in-line coding (e.g. data statements) or better by an on-line interpretation of data description tables. The description information refers to Parameter Names (like variable names in a programme) which associates the Parameter Type, the position within the structure, and its interpretation (e.g. unit, calibration, scaling, attached procedure etc.).

The Parameter Type defines a class of encoding of the parameter value. For a given Parameter Type there are

possible variations in the length of the value encoding. The only Parameter Types allowable for Telemetry and Telecommand parameters are those listed in Table 3.

PARAMETER TYPE	LENGTH	DEFINITION
Boolean	1 bit	0 = False , 1 = True
Logical	8/16 bits	Used in logical operations
Integer	1/2/4/8 octets	Signed or unsigned (ISO)
Real	2/4/6/8 octets	(ISO)
String	0 - 255 octets	ASCII (ISO) character string
Time	1 - 8 octets	CCSDS time (CUC or CDS)
Address	2/4 octets	Offset, base or offset + base

Table 3 : List of Parameter Types

A Record is a logical grouping of different Parameters or Arrays or other Records. An Array contains several instances of the same Record. By this definition the Packet Data Field is a Record. Consequently there could be several levels of nesting of Arrays and Records, although this flexibility should not be abused.

An Array can either be a Fixed-Array of a fixed number of instances (known implicitly by the application according the structure definition) or be a Variable-Array of a variable number of instances indicated in front of the Array. The structuration rules are presented in a "syntax diagram" form on Figure 5.

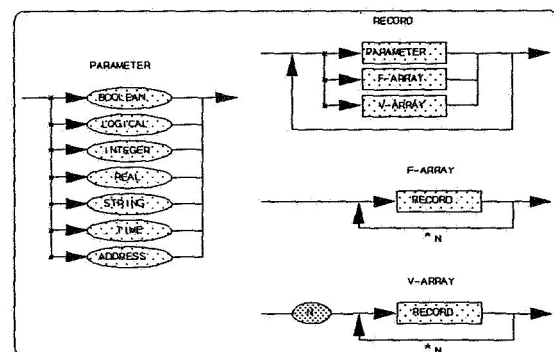


Fig. 5 : Data Field Structure Rules

8. VALIDATION

Before approving this standard, and before implementing supporting infrastructures, it was necessary to ensure its correctness, its practicability and operational usefulness. This has been achieved by a prototyping exercise completed in early 1992, which has validated the standard and at the same time given some indication on the implementation technique.

The Packet communication technique has not been addressed in this prototype since this is already demonstrated. Rather the application end-to-end aspect was implemented, emulating the Spacecraft Application behaviour in relationship to the Ground Application (e.g. control system) in control of it.

This prototype (called PUSV) runs on one or two SPARC workstations and permits at the same time to model different on-board Application architectures. A reference satellite model (called PUSSAT) was implemented for demonstration purposes.

9. FUTURE PERSPECTIVES

The PUS draft standard will be reviewed at Agency level and it is expected that it will become an Agency approved standard in the first half of 1993.

Furthermore, this standard is expected to evolve in the future in an incremental way when some commonly used monitoring and control functions are mature enough to be generalised and thus standardised.

ESA plans to develop the necessary ground infrastructures (e.g. mission control kernel system) to support this standard but also more generally the operational functions of such a system in order to achieve in reality the vertical and horizontal commonality of a generic system to be used for checkout and operation across different projects.

10. REFERENCES

1. (ESA) Packet Utilisation Standard (PUS), STD-07-101/92, issue February 1991
2. (ESA) Packet Telemetry Standard, PSS-04-106, issue 1 January 1988
3. (ESA) Packet Telecommand Standard, PSS-04-107, issue 2 April 1992
4. (CCSDS) Packet Telemetry 102.0-B.2, Blue Book January 1987
5. (CCSDS) Telecommand: Part 3, Data Routing Service, 203.0-B.1, Blue Book January 1987
6. (ESA) EGSE & Mission Control System (EMCS) Functional Requirements Specification PSS-07-401, 7 September 1992

MULTIMISSION IMAGE PROCESSING AND SCIENCE DATA VISUALIZATION

N94-23955

William B. Green

California Institute of Technology
Jet Propulsion Laboratory
Pasadena, CA

ABSTRACT

The Operational Science Analysis (OSA) Functional area supports science instrument data display, analysis, visualization and photoprocessing in support of flight operations of planetary spacecraft managed by the Jet Propulsion Laboratory (JPL). This paper describes the data products generated by the OSA functional area, and the current computer system used to generate these data products. The objectives on a system upgrade now in process are described. The design approach to development of the new system are reviewed, including use of the Unix operating system and X-Window display standards to provide platform independence, portability, and modularity within the new system, is reviewed. The new system should provide a modular and scaleable capability supporting a variety of future missions at JPL.

1. FUNCTION AND ROLE OF THE OPERATIONAL SCIENCE ANALYSIS FUNCTIONAL AREA

The OSA Functional Area is responsible for providing a multimission capability for science instrument data display and analysis supporting flight operations. In support of that role, the functional area provides real and near-real time display of science instrument data as it is received, and provides hardware and software to support data analysis. The OSA functional area also produces archival data products, including digital data records for individual instruments that are transferred to the Planetary Data System, archival quality photographic products, and catalogs describing the content of all data products that are generated.

2. DATA PRODUCT EXAMPLES

Examples of recent data products generated by the functional area include the following:

- Digital image mosaic data products of the surface of Venus generated by merging Synthetic Aperture

Radar (SAR) image data acquired from many orbits into rectangular mosaics corresponding to segments of the surface indexed by longitude and latitude. These products were produced as photographic hardcopy and as digital data records produced on CD-ROM media.

- Digital image data products of the Earth and Moon generated from data returned by the solid state imaging (SSI) camera on the Galileo spacecraft. These products were also produced in cataloged digital format on CD-ROM media and as black and white and color photoproducts.

- Photographic products of the Earth and Moon generated from data returned by the near-infrared mapping spectrometer (NIMS) instrument on the Galileo spacecraft. This instrument returns up to 256 spectral bands of information for selected regions of the surface. Special photoproducts are generated showing a false color rendition of the full multispectral image, false color histograms depicting the radiance across all spectral bands, and selected spectra for six different points on the surface depicted as line plots. Digital data records for this instrument are produced on magnetic tape.

- Photographic representation of spectra derived from the plasma wave subsystem (PWS) depicted as a series of false color photographic data products.

These examples indicate the flexibility of the same data processing system to producing a variety of different products for more than one mission and more than one instrument type.

3. CURRENT HARDWARE SYSTEM

The current system configuration is shown in Figure 1. The system consists of four Digital Equipment Corporation VAX computer systems and a variety of peripheral devices. The VAX systems are configured in a cluster configuration, so that the four processors have independent access to shared magnetic disk storage. Approximately 100 GB of magnetic storage

are available on the system, reflecting the large storage volumes required to handle the quantity of digital image data returned by the planetary missions. The magnetic storage is augmented by optical disk storage as shown. 64 GB of optical disk storage is available on two jukebox systems.

The system is configured with special purpose peripheral devices designed to support data display and visualization. Dedicated image displays support viewing and manipulation of 24 bit image data and associated graphics overlays at resolutions up to 1024 x 1024 picture elements. Black and white and color film recorders provide high precision film recording on 9 inch wide film at spot sizes starting at 12.5 microns.

4. CURRENT SOFTWARE SYSTEM

The system shown in Figure 1 currently operates under the DEC VMS operating system. JPL's VICAR image processing software system is used for the majority of applications supported by this equipment. VICAR runs under an executive called TAE (Transportable Applications Executive) developed at Goddard Space Flight Center. TAE provides the main user interface with the software system. Over 200 individual applications programs are currently maintained as part of the system. A general purpose subroutine library supports specialized high performance requirements associated with image and signal processing.

Use of a standard image format within the system insures that image and science data acquired from a variety of sources can all be processed by the same applications software, once the data have been converted from the original format to VICAR format.

A device independent approach has been taken to support image display on a variety of image display equipment provided by different vendors. The device dependent display and manipulation functions are isolated from the applications programmer; individual programs call a general set of display routines that handle the specific capabilities and characteristics of the hardware displays actually used by analysts in executing the software.

An interface with the Datatreive data base management system is maintained, so that a record is maintained of all data processed by the system. The catalogs generated using Datatreive also include ancillary information relating to the science instrument data (e.g., navigation data, engineering data, etc.).

5. SYSTEM UPGRADE OBJECTIVES

The system shown in Figure 1 was developed starting approximately 10 years ago, and has supported the Voyager, Magellan and Galileo missions. A complete upgrade of this system is planned in time to support Galileo operations at Jupiter. The objectives of the new system design include the following:

- Eliminate dependence on a single vendor for computing platforms.
- Provide a modular scaleable design that can accommodate the variety of missions now under consideration
- Develop a standards complaint system, providing portability and ease of data transfer and exchange
- Maintain the multimission and multiinstrument nature of the current facility
- Provide commercially available visualization software and other commercially available software to minimize in-house development resources required to address new applications

6. SYSTEM UPGRADE APPROACH-- SOFTWARE

The basic software approach that is being taken in the redesign of the system can be summarized as follows:

- Utilize the Unix operating system to provide hardware independence
- Convert the existing VICAR applications capability to Unix, to preserve almost 20 years of investment in a highly flexible multi-instrument and multimission capability
- Utilize the Sybase data base management system, providing a state of the art relational data base capability
- Utilize X-Windows as a standards-complaint image display and manipulation environment, providing vendor independence in selection of high resolution display equipment
- Utilize commercially available visualization packages operating under TAE to augment the VICAR capabilities for data visualization as required

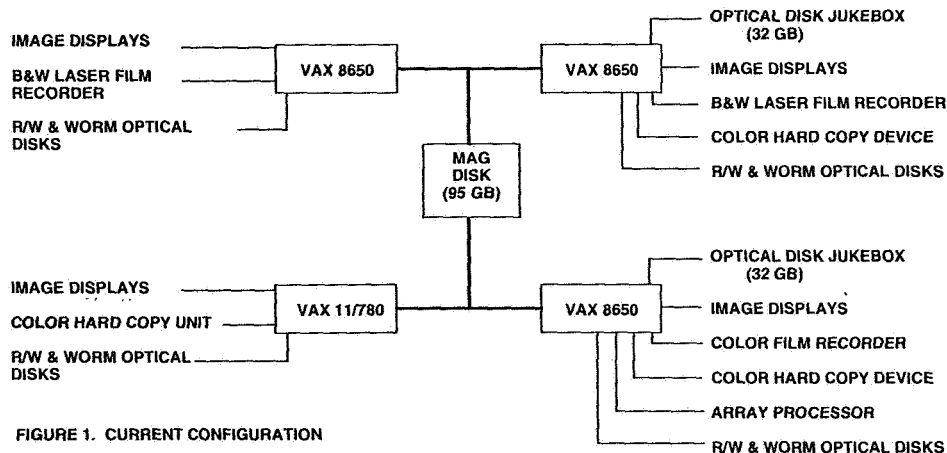
7. SYSTEM UPGRADE APPROACH-- HARDWARE

The hardware approach being taken in the redesign effort can be summarized as follows:

- Utilize Unix based workstations as the basic modular building blocks within a distributed design
- Support data management and file serving from dedicated processors
- Utilize X-Window compliant workstations to support image display functions, establishing vendor independence and portability
- Utilize FDDI as a basic communications backbone within the new system, providing the highest available speed for transfer of image and signal data between nodes

8. SUMMARY

The OSA functional area has a long history of developing and implementing a flexible multimission capability to support production of data products supporting flight operations at a minimal development and operations cost. Technology developments in the ten years since the last major upgrade of the OSA system now make it possible to achieve a standards-compliant modular new design that can be adapted to support multiple missions and different instruments at minimal development cost. Use of a modular design and implementation approach, and utilization of a new generation of standards-compliant hardware systems should also reduce the operations cost associated with support of individual missions.



N 9 4 - 2 3 9 5 6

COMPRESSION FOR AN EFFECTIVE MANAGEMENT OF TELEMETRY DATA¹

J.-P. Arcangeli*
 M. Crochemore**
 J.-N. Hourcagnou ***
 J.-E. Pin****

*Institut de Recherche en Informatique de Toulouse & Greco de Programmation du C.N.R.S.,
 Université P. Sabatier, 118 Route de Narbonne, 31062 Toulouse Cedex, France

**Laboratoire d'Informatique Théorique et Programmation & Greco de Programmation du C.N.R.S.,
 Université Paris VII, 2 Place Jussieu, 75251 Paris Cedex 05, France

***Centre National d'Etudes Spatiales, TE/IS/PS/TD,
 18 Avenue E. Belin, 31055 Toulouse Cedex, France

****Laboratoire d'Informatique Théorique et Programmation & Greco de Programmation du C.N.R.S.,
 Université Paris VI, 4 Place Jussieu, 75252 Paris Cedex 05, France

ABSTRACT

A Technological DataBase (T.D.B.) records all the values taken by the physical on-board parameters of a satellite since launch time. The amount of temporal data is very large (about 15 Gbytes for the satellite TDF1) and an efficient system must allow users to have a fast access to any value. This paper presents a new solution for T.D.B. management. The main feature of our new approach is the use of lossless data compression methods. Several parametrizable data compression algorithms based on substitution, relative difference and run-length encoding are available. Each of them is dedicated to a specific type of variation of the parameters' values. For each parameter, an analysis of stability is performed at decommutation time, and then the best method is chosen and run. A prototype intended to process different sorts of satellites has been developed. Its performances are well beyond the requirements and prove that data compression is both time and space efficient. For instance, the amount of data for TDF1 has been reduced to 1.05 Gbytes (compression ratio is 1/13) and access time for a typical query has been reduced from 975 seconds to 14 seconds.

Key Words : data compression, technological database, temporal data.

1. INTRODUCTION

The paper describes a C.N.E.S. study which provides new solutions for the management of technological databases. A Technological DataBase (T.D.B.) contains all the technological telemetry data downlinked from the satellite. Technological telemetry data are values of physical parameters (temperatures, pressures, ...) measured by the on-board platform sensors. They serve to control the satellite, and to follow the performance of on-board equipments. Their storage in a T.D.B. raises two major difficulties :

- as data are produced at a high rate, space needed for storage is very large (for a telediffusion satellite like TDF1, the amount of data exceeds 15 Gbytes at the end of the satellite's lifetime) ;
- in critical cases, when failures occur on board, satellite specialists need efficient access to all data recorded since launch time, and, more generally, users do not need access to one single value, but to a large set of values (e.g., all the values taken by a parameter during one week or one month).

Until now, the design of the T.D.B. was based on the following concept : the telemetry frames are stored in a single file without any transformation. This strategy has two major defaults :

¹ supported by C.N.E.S. and GRECO de Programmation du C.N.R.S.

- due to the large amount of data, one has to archive the oldest periods on magnetic tapes ;
- it does not provide efficient access to data.

In 1990, the C.N.E.S. decided to study new solutions for T.D.B. management. The scope of this study was to design and produce a prototype of T.D.B. fulfilling the following requirements :

- the prototype would be able to adapt to different satellites' telemetries and to process real data from geostationary (e.g. TDF1) or polar orbit satellites (e.g. SPOT1) ; so, it would be based on the usage of dictionaries which describe the telemetry format ;
- it would implement data compression methods which guarantee perfect restitution of the information (no bit lost) and cover different types of parameter evolution (some very irregular, others very stable) and different categories of parameter characteristics (length, frequency) ;
- it would provide fast access to all data ;
- it would detect and manage the lack of telemetry lines due to transmission failures.

This prototype has been successfully developed in collaboration with GRECO Informatique du C.N.R.S. and its performances are well beyond the requirements. For instance, the total amount of technological data has been reduced from 15 Gb. to 1.05 Gb. thanks to lossless data compression methods. Extraction of one parameter over 2 weeks (150000 values) which is 975 seconds long in a "traditional" T.D.B. on a CDC 990, is only 14 seconds long with the prototype running on SUN 4/330.

The major principles of our software architecture are as follows. Before storing the final values in the T.D.B., the telemetry frames are decommutated and data corresponding to a parameter are collected on a one-day (or one-pass) basis. Then, for each parameter, a dynamic choice of the most appropriate compression method is made, depending on the stability of the parameter's values. These values are compressed and stored in a specific file (dedicated to the parameter considered) with one block per processed day (or pass). So, during the extraction phase for each requested parameter, we have only to select the blocks corresponding to the requested

period by means of a global index, and to decompress these ones.

Section 2 explains how temporal data are represented in our data organization. Section 3 gives a general description of the lossless data compression methods, and section 4 focuses on the specific compression techniques which have been implemented here. Physical organization and software processing are described in section 5. Finally, experimental results are provided in section 6.

2. TIME MANAGEMENT

Parameters have a specific bit-length and frequency. They are measured in a synchronous way. Values are temporal data sent into fixed structured packages called telemetry frames. Frames are composed by telemetry lines which have a regular frequency (one second). All the data of one line are measured at the same time. Transmission of telemetry lines is all the day long for geostationary satellites, but it is gathered in passes for polar orbit satellites (e.g., 8-10 passes per day of 10-12 minutes for SPOT1).

To be efficient, temporal data must first be grouped by parameter : *decommutation* of telemetry frames is a basic principle. For any processed period, it produces a set of values for each parameter. The date of a value can be expressed as a function of the initial date, of the parameter's frequency, and of the range of the value in its set. This strategy leads us to an all-parameter implicit time representation, but we must deal with missing telemetry lines.

The lack of lines can be located and its duration calculated. Rather than replacing a missing value by a special code (this is not easy in particular for one-bit parameters nor is it space efficient as encoding a lack of lines is repeated for any affected parameter), we build a descriptor of the processed period like this :

number of correct line(s), number of missing line(s), number of correct line(s), number of missing line(s) ...

Such a descriptor enables us to find the date of any telemetry line and thus of any parameter value. Hence, decommutation produces a time-

ordered sequence of values (where missing ones do not appear) for every parameter and an all-parameter time description.

3. LOSSLESS DATA COMPRESSION METHODS

We are only interested in lossless data compression methods, also called text compression methods. The reduction in size of the source data is obtained by special encodings applied on them. A corresponding decoding phase is afterwards applied, when needed, to restore the original data. The lossless feature remains to say that the two phases, encoding and decoding, applied in that order to any input makes absolutely no change on the data.

The main interest for compression methods is both for file archiving, and for transmitting data. It is usually thought that compressing slows down the access to data. This is not entirely true in telecommunications because this somehow depends on the speed of the channel relatively to the time spent in encoding and decoding the data. The success of Facsimile machines is likely to be credited to text compression. The present article still provides another example which shows that both space and access time can be improved simultaneously.

It is possible to classify text compression methods according to their probable efficiency. But the performance also depends on the data. More precisely, it depends on the *entropy* of the source, which tells us how regular are the data. If the data are random, the entropy is high, and no method can compress them efficiently.

In this section we describe general data compression methods. Since methods are general, they are purely syntactic. No semantic data compression method is considered. So, compression ratios, which are *ratios between the volume of compressed data and the original volume*, must be appreciated under that condition. Methods commonly save about 50% memory space.

Text compression methods are mainly based on substitutions. It can be considered that the source s , and the encoded text c , are strings over the alphabet $\{0, 1\}$. The source s is often itself

the encoding of a text written over some more natural alphabet. For instance, s can be the concatenation of ASCII codewords if it is the translation of a text in natural language. We call t the text form which s is translated, and A the alphabet of t . The translation is then described by a function f , which gives the (unique) translation of each letter. Then, defining an encoding remains to specify another function h from A to $\{0, 1\}^*$. The encoding of s is then the translation of t by the function h . The set $\{(f(a), h(a)) / a \in A\}$, or simply $\{(a, h(a)) / a \in A\}$ if function f is implicit, is called the *dictionary* of the compression method. Functions f and h extend to morphisms from A^* to $\{0, 1\}^*$. The lossless condition is satisfied if h , as a morphism, is one-to-one, which means that the set $\{h(a) / a \in A\}$ is a (*variable length*) code (Ref. 1).

The pair of functions, (f, h) , leads to a classification of data compression methods based on substitutions. We get four principal classes considering both whether f is uniform (i.e. when all images of letters are words of the same length) or not, and whether the dictionary is fixed or computed during the compression process. Most elementary methods do not actually use any dictionary.

	uniform morphism	non uniform
	relative or topographic encoding	repetition encoding
fixed dictionary	statistical encoding (Huffman)	factor encoding
evolutive dictionary	sequential statistical encoding (Faller and Gallager)	Ziv and Lempel's algorithm

The first line of the above array mentions elementary methods. For suitable data they can give excellent compression ratios. These methods have been eventually chosen to encode the telemetry data.

Statistical encodings are among the most popular methods. The Unix (system V) command "pack" implement the famous

Huffman's algorithm (Ref. 2). It admits a sequential version, discovered independently by Faller (Ref. 3) and Gallager (Ref. 4), well suited for communication, and implemented by the "compact" command of Unix (BSD 4.2). Tests with these methods applied to telemetry data has shown that the process is too general to compress them efficiently.

Finally the last method we have tried is factor encoding. It contains the sequential algorithm of Ziv and Lempel (Ref. 5) on which the "compress" command of Unix (BSD 4.2) is based. This method often gives the best compression ratio on ordinary data. It served as a reference to evaluate the entropy of the sources.

An extensive presentation of text compression methods can be found in Ref. 6.

4. DATA COMPRESSION TECHNIQUES

We have used four different data compression techniques DCT1, DCT2, DCT3 and DCT4. These techniques are adaptative methods based on substitution encoding, relative encoding and run-length encoding. They require data preprocessing in order to optimize the choices of certain parameters in our algorithms. In practice, in order to save time, we have incorporated this analysis into the decommutation process. That is, the data is first analyzed to find which of our compression schemes and which values of the parameters will produce the best compression ratio, and are then compressed using this best method. We now describe in detail our four data compression techniques.

4.1. DCT1

Some parameters only have a very restricted number of values (say 12 for instance). It is therefore possible to assign a special 4-bit code for these 12 values. If there were only 6 different values, one would have used a 3-bit encoding. Thus, the number of bits used for this uniform coding is a parameter of this method. Note that it is however necessary to have a special code to indicate exceptional values. For instance, if the 6 normal values are 121, 125, 129, 133, 137 and 141, one would use the following coding

121 → 000	125 → 001	129 → 010
133 → 011	137 → 100	141 → 101

Here, 110 is unused and 111 can indicate that the next n bits represent an exceptional value. Of course, the dictionary should be memorized. This method can be completed by a run-length algorithm.

4.2. DCT2

This method is a simple relative encoding. For instance, let

117 121 122 117 121 128 121

be a sequence of successive values of a certain 8-bit parameter. By relative encoding, each value other than the first is coded with the relative difference between it and the preceding value. In our example, we obtain

117 +4 +1 -5 +4 +7 -7

The trick is now to use an optimal bit representation to code the differences. In our example, the extremal relative differences are -7 and 7, which gives an interval of 15 possible values. It is therefore possible to code all the differences by a 4-bit representation. For instance, one can use the leftmost of the four bits to code the sign (1 for + and 0 for -) and the three remaining bits to code the absolute value of the difference. This gives the following code

0 → 0000	1 → 1001	-1 → 0001
2 → 1010	-2 → 0010	3 → 1011
4 → 1100	-4 → 0100	5 → 1101
6 → 1110	-6 → 0110	7 → 1111
		-7 → 0111

Now, since the 8-bit representation of 117 is 01110101, we obtain the following encoding of our sequence

01110101 1100 1001 0101 1100 1111 0111
117 +4 +1 -5 +4 +7 -7

with a compression ratio of $4/7 \cong 57\%$...

If the extremal relative differences were for instance -2 and +3, we would have coded these differences by a 3-bit representation only. Determination of the optimal representation

takes place during data preprocessing mentioned above.

DCT2 is especially efficient on data with few repetitions but small relative differences.

4.3. DCT3

This third method is a run-length encoding used for very regular data, that is, data containing a large proportion of repeated values and relative differences almost always equal to 1 or -1. We describe the algorithm with an example. Consider the following sequence of values of an 8-bit parameter :

90 90 90 91 91 91 91 91 91 91 91 91 91 91
91 91 91 90 90 90 91 91 91 91 91 91 91 91
91 91 91 91 91 91 91 91 91 91 91 91 91 91
91 91 91 91 91 91 91

This sequence can be synthesized as follows

90 (x 3), 91 (x 15), 90 (x 3), 91 (x 31)

or, after relative encoding, by

90 (x 3), +1 (x 15), -1 (x 3), +1 (x 31)

Now the coding becomes easy. The first byte codes the first value (90 on our example). The second byte codes the number of repetition of the first value. The next bytes are decomposed as follows : the leftmost bit encodes the relative difference (1 for +1 and 0 for -1) and the other bits encode the number of repetitions. In our example, we obtain

01011010 00000011 10001111 00000011
90 3 +1 (x 15) -1 (x 3)
10011111
+1 (x 31)

with a compression ratio of $5/52 \cong 9.6\%$...

There are two obvious problems with this encoding. First, the number of repetitions is limited to 127, the biggest 7-bit number. Second, the coding has to be changed if one of the relative differences is different from 1 or -1. However, it would be convenient to use the same scheme for data containing only exceptional values for which the relative differences exceed one in absolute value.

The first problem can be easily avoided. It suffices to determine the maximal number of repetitions during data preprocessing and code the number of repetitions accordingly. For instance, if the maximal number of repetitions is 56, a 6-bit representation will be enough. If it is 57,243, a 16-bit representation will be required.

Assume now that a n-bit representation is used for the number of repetitions. Then a typical encoding for an 8-bit parameter will have the following form

First value	First number of repetitions.
0 / 1	number of repetitions
0 / 1	number of repetitions
0 / 1	number of repetitions ...

on 8 bits + n bits + 1 bit + n bits + 1 bit + n bits + 1 bit + n bits + ...

Since there is at least one repetition, the codes

0 00 ... 0 and 1 00 ... 0

on n bits are never used. This gives the solution to our second problem. If an exceptional relative difference is different from 1 and -1, we use the sequence (on n + 8 + n bits)

0 00 ... 0 relative difference
number of repetitions

The code 1 00 ... 0 on n bits is never used, but is saved for future use if the software needs a further development.

4.4. DCT4

Our fourth method combines the ideas of the two previous methods. That is, data is first encoded by relative encoding and then a run-length encoding is applied. As usual, the maximal relative differences and the maximum number of repetitions are determined during the decommutation pass.

Assume for instance that the maximal relative differences are - 3 and + 2 (giving a range of 6 possible values) and that the maximum number of repetitions is 13. Then it is possible to use a 3-bit representation for the differences and a 4-bit representation for the repetitions. Consider for instance the following sequence of values of an 8-bit parameter

```
90 90 92 92 92 89 91 91 91 91 93 93 93 93
93 93 93 93 93 94 94 94 94 94 94 92 92 92
92 92 92 92 92 92
```

By relative encoding, we obtain the sequence

```
90 0 +2 0 0 -3 +2 0 0 0 0 +2 0 0 0 0 0 0 +1
0 0 0 0 0 0 -2 0 0 0 0 0 0 0 0
```

Then, we can encode the repetitions by run-length encoding. In practice, it is convenient to keep a special code to indicate that we start run-length encoding. Thus, in our example, we would use the following code for the differences

```
- 3 → 110      - 2 → 101      - 1 → 100
0 → 000        + 1 → 001      + 2 → 010
```

and keep the code 111 to indicate that the next four bits will represent the number of repetitions n . Notice that if a value occurs two or three times, it is not efficient to encode the number of repetitions : for instance, for the three consecutive 92, the encoding 010 000 000 will be shorter than 010 111 0011. For this reason, we will only encode repetitions if $n \geq 4$. Consequently, a possible improvement would be to encode $n-4$ instead of the number n of repetitions. If we don't use this improvement, we obtain for our example the following encoding

```
01011010 000 010 000 000 110 010 111 0101
010 111 1001 001 111 0111 101 111 1010
```

which codes, literally

```
90 0 +2 0 0 -3 +2 (x 5) +2 (x 9) +1 (x 7) -2 (x 10)
```

leading to a compression ratio of $63 / 296 \approx 21 \%$

5. PHYSICAL ORGANIZATION AND SOFTWARE PROCESSING

After decommutation is done, any single-parameter sequence of values is compressed by the chosen specific compression technique. The algorithm number and its parameters are associated with the compressed sequence to create a block. Blocks, whose sizes are variable, are clustered into single-parameter files. For their location, a two-entries index table (the parameter and the processed period numbers) is updated with every parameter's compressed data block address and size in the file, and then stored in an index file.

The choice of the database updating time unit must be solved with regard to the expected time and space efficiency. In reality, time segmentation must fit the users requirements, knowing that volumes must be sufficient for the compression to be significant, but not too large to avoid the worst cases decreasing general efficiency. It should be noted that, using the previous techniques, updating units are also extraction ones since the entire block must be processed at decompression time. In our experiments, we processed one-day periods for TDF1 and only passes (10-15 minutes) for SPOT1.

In such a system, an any-length multiparameter query is first divided in function of the sub-periods in the database using the time descriptor file, and then is decomposed by parameter. Each block is located using the index file and then read from the parameter file. The sequence of values is decompressed according to the algorithm references set on the block. Finally, extracted values are dated using the time descriptor file and the parameter dictionary.

6. EXPERIMENTAL RESULTS

A modular parameterizable prototype has been developed for TDF1 telemetry (Ref. 7). Another one, intended to process SPOT1 telemetry, was easily built (Ref. 8) from the first one. These prototypes are C programs running on Unix system.

Experimental results presented correspond to programs compiled with the option -O (simple

optimization), and run on SUN4 Sparc 330 for TDF1 and IBM RS6000 520 for SPOT1 (RS6000/520 is 1.5-2 faster than SUN4/330). Given times are user + system in seconds.

6.1. TDF1.

The first table contains compression results in volume and time for one-day sequences of values

parameter	origin. volume	DCT1	DCT2	DCT3	DCT4	compress	chosen DCT	compression ratio	compression time	decompression time
A301K 8 bits	2696 bytes	1359 b.	678 b.	137 b.	94 b.	354 b.	4	3.5%	0.03	0.00
A605G 10 bits	216 Kb.	195 Kb.	69 Kb.	207 Kb.	69 Kb.	115 Kb.	4	31.8%	3.59	0.92
A6083 10 bits	216 Kb.	108 Kb.	73 Kb.	215 Kb.	74 Kb.	91 Kb.	2	33.7%	2.22	0.86
A673 16 bits	5392 b.	338 b.	679 b.	6 b.	8 b.	122 b.	3	0.1%	0.04	0.01
D403H 8 bits	2697 b.	2953 b.	692 b.	2702 b.	694 b.	1322 b.	2	25.7%	0.06	0.02
P305K 8 bits	2696 b.	1714 b.	678 b.	182 b.	250 b.	554 b.	3	6.8%	0.02	0.00
R101Z 1 bit	5392 b.	—	11 Kb.	—	7 b.	123 b.	4	0.1%	0.22	0.04

Database updating was experimented with 50 days of real telemetry (50 days is about 2.5% of the satellite life span estimated at 2000 days). The original one-day volume is nearly 7 Mb. but after compression the average amount of data is 520 Kb. only (add 7 Kb. for management files). Thus, the average compression ratio is approximately 8% (day-ratios go from 7.44% to 9.93%). This would lead to 1 Gb. of compressed data for 2000 days, instead of 14 Gb.

original volume	compressed volume	compression ratio
7.2 Mb.	0.52 Mb.	8 %

Duration of a one-day updating is nearly 6 minutes (3 minutes 1/2 only on IBM RS6000 520).

decommutation	compression	total time
306.1	51.4	357.5

of some representative parameters. Volumes produced by the Unix "compress" command are also indicated. Considering the produced volumes, DCT1 has not been implemented for TDF1. Remark that any one-day block of values is decompressed in less than one second.

Finally, consider two unfavorable selections of 5 and 20 parameters and note the response times for queries over several period lengths.

period length	decompressed blocks	5 param.	20 param.
one hour	1	4	10
one day	2	11	34
one week	8	51	185

6.2. SPOT1.

Using DCT1 is interesting for compressing some SPOT1 parameters : total volume is decreased by about 20%. We show a few significative examples.

parameter	original volume	best DCT and ratio	DCT4 ratio
SCCOU3	727 b.	- >100%	76 %
SCECR2	1292 b.	- >100%	47 %
SVROU3	1292 b.	2 49%	27 %
SPHI4	1292 b.	3 66%	27 %
SXTHEx	1292 b.	3 88%	34 %

The next table summarizes time and space results when updating the database with one medium 10-12 minutes length pass.

average original volume	average compres. volume	average compres. ratio	total time
216 Kb.	27 Kb.	13 %	8.4

As a day is composed by 8 passes, one-day amount of data is closed to 220 Kb. instead of 1.75 Mb., and one minute only is necessary to a complete one-day updating. Finally, note that grouping one-day passes and updating the by day would produce an average volume of 199 Kb. and so decrease the volume by about 10%.

7. CONCLUSION

We are currently developing a new operationnal T.D.B. for the satellites TDF1 and TDF2 which is based on this new architecture and linked with a software package for visual data analysis. This new T.B.D. will be an efficient tool to follow the performance of on-board equipments.

8. ACKNOWLEDGEMENTS

The authors would like to thank Michel Mouyssinat whose intuition and enthusiasm led to this fruitful collaboration between C.N.E.S. and GRECO Informatique du C.N.R.S.

9. REFERENCES

1. Berstel, J., and Perrin, D. 1985. *Theory of codes*. Academic Press.
2. Huffman, D.A. 1951. A method for the construction of minimum redundancy codes. In *Proc. IRE* 40, 1098-1101.
3. Faller, N. 1973. An adaptive system for data compression. In *Record of the 7th Asilomar*

Conference on Circuits, Systems, and Computers, 593-597.

4. Gallager, R.G. 1978. Variations on a theme by Huffman. In *I.E.E.E. Trans. Inform. Theory* IT 24, 6, 668-674.

5. Ziv, J., and Lempel, A. 1978. Compression of Individual Sequences via Variable-rate Coding. In *I.E.E.E. Trans. Inform. Theory* IT 24, 5, 530-536.

6. Bell, T.C., Cleary, J.G., and Witten, I.H. 1990. *Text compression*. Prentice Hall.

7. Arcangeli, J.-P., Crochemore, M., and Pin, J.-E. 1991. Rapports d'études préalables, de conception, d'évaluation et manuels d'installation et d'utilisation du prototype de banque de données technologiques TDF1. Centre National d'Etudes Spatiales, Toulouse, France.

8. Roux, L. 1992. Prototype de banque de données technologiques SPOT1. Université P. Sabatier et Centre National d'Etudes Spatiales, Toulouse, France.

Space Station Freedom Ground Data System: **N 9 4 - 2 3 9 5 7** Design and Operations

Richard A. Dunning, Jr.
Booz-Allen & Hamilton Inc.
Washington, DC

Stanley A. Fishkind
NASA Headquarters
Washington, DC

Frederick W. Knops, III
Booz-Allen & Hamilton Inc.
Seabrook, MD

Michael P. Pasciuto
NASA Headquarters
Washington, DC

ABSTRACT

Over the previous year the Space Station Freedom (SSF) Program (SSFP) ground data distribution system has become independent of a number of data systems that were to have been provided by other National Aeronautics and Space Administration (NASA) programs. Consequently, the SSFP has outlined the basic architecture of a new data system dedicated to supporting SSF requirements. This has been accomplished through a complete redesign of the ground network and a reallocation of selected functions.

There are a number of aspects of the new ground data distribution system that are unique among NASA programs. These considerations make SSF ground data distribution one of the most extensive and complex data management challenges encountered in the arena of Space Operations. A description of this system comprises the main focus of the paper.

Key words: Space Station Freedom, data system, operations

1. INTRODUCTION

The SSFP is an international endeavor with space and ground-based elements contributed by the United States through NASA, the European community through the European Space Agency (ESA), Japan through the National Aerospace Development Agency (NASDA) and Canada through the Canadian Space Agency (CSA).

The Space Station Manned Base (SSMB) will be a multipurpose orbiting facility providing the means for a permanent human presence in space and the pursuit of scientific endeavors in areas such as microgravity and materials sciences, astrophysics, human physiology, and earth sciences. Experiments may be conducted either inside the pressurized modules or exposed to space at truss attach points.

Beginning in 1999, Freedom will provide the capability for permanent human habitation during the Permanently Manned Capability (PMC) phase. Prior to this stage, a Manned Tended Capability (MTC) is planned. In this intermediate phase of construction, temporary habitation and experimentation will occur during visits by the space shuttle crew. MTC will begin with the delivery of the first pressurized module in 1997. Prior to that time, the truss "infrastructure" necessary to support the pressurized modules with power, communications, and other utilities will be delivered and assembled in space.

The ground data distribution system planned by the SSFP will be developed over the same time period and will continue to evolve during the 30-year orbital life of the SSMB. Initial capabilities, in place about the time of the first delivery mission in 1996, must ensure the safe operation of the systems on orbit. The initial scientific use of the SSF in 1997 will require a far more extensive and capable ground network to support the investigations of the international scientific communities participating in the program.

It is important to note that the ground data processing and data distribution functions are the most recent major additions to the SSFP. Prior to 1991, two programs were under way within NASA to provide these services centrally to all NASA space programs, including unmanned satellites. In that year, these programs, NASA Communications II (Nascom II) for data transport and the Customer Data Operations System (CDOS) for data processing, were eliminated from NASA due to budget considerations, and the responsibilities for data services required by the Earth Observing System and SSFP were transferred to those programs. The SSFP has maintained much of the data-driven architecture envisioned by the CDOS and Nascom II programs, but has tailored the implementation to suit the specific needs of the program through a complete redesign of the ground network and a reallocation of selected functions.

2. UNIQUE REQUIREMENTS

As currently projected, the primary data volume driver for SSF data operations will be the science data downlink. Operation of the SSF as a science laboratory will begin in earnest upon the arrival of the first complement of payloads in 1997. A variety of payloads are already scheduled for the MTC period, but primary emphasis will be on microgravity and life-sciences payloads. These areas of research place distinct requirements on the SSFP ground-based data systems. Life-sciences payloads will rely heavily on video, and its correspondingly high data rate, for information collection. Microgravity science is sensor-data-intensive and depends on real-time interaction between the payload and ground-based operators for payload operations. The dual nature of these requirements — large data rates and real-time operations — provides a challenging environment for SSFP ground operations.

Further demands result from the international nature of the Program. International and joint payloads require the distribution of downlinked data to the International Partners (IPs), often with the same real-time requirements of U.S. payloads. Uplink data (both core and payload) destined for IP

modules will be given the same priority as U.S. uplink data, but must first pass through source authentication processes.

An additional, and perhaps most challenging aspect of SSFP communications is the requirement to allow continuously expanding data rates over the 30-year life of the program. Unlike previous long duration missions, such as Voyager, the expansion of the SSMB and evolution of its contingent of payloads will allow data requirements to increase over the life of the program. Developments such as the deployment of the next generation of tracking and data relay satellites at the end of the century will make rapidly expanded data rates feasible — and likely.

3. GROUND DATA DISTRIBUTION ARCHITECTURE

The figure on the following page presents the SSFP ground data distribution architecture, which consists of the elements listed below. A distinction is made between payload data and core data (i.e., SSMB system data).

SSMB The Space Station Manned Base is the free-flying space station element.

TDRSS The Tracking and Data Relay Satellite (TDRS) System (TDRSS) is a constellation of geosynchronous NASA communications satellites designed to support low earth orbit science missions.

WSC The White Sands Complex, located near Las Cruces, New Mexico, is the site of the TDRSS ground terminal. WSC provides the sole capability for TDRS uplink and downlink connectivity.

SSCC The Space Station Control Center, located at the Johnson Space Center (JSC) in Houston, Texas, will be the focal point for SSMB operations. The SSCC will perform the following functions:

- Command and control of the SSMB
- Management and integration of core operations and core operations planning
- Interface with Engineering Support Centers (ESCs) for maintenance and sustaining engineering support

- Interface with the POIC to receive payload commands for uplink to the SSMB
- S- and Ku-band data capture
- Core data processing, archiving, and distribution
- Video processing and distribution.

POIC The Payload Operations and Integration Center, located at Marshall Space Flight Center (MSFC) in Huntsville, Alabama, will be the focal point for payload operations. The POIC will perform the following functions:

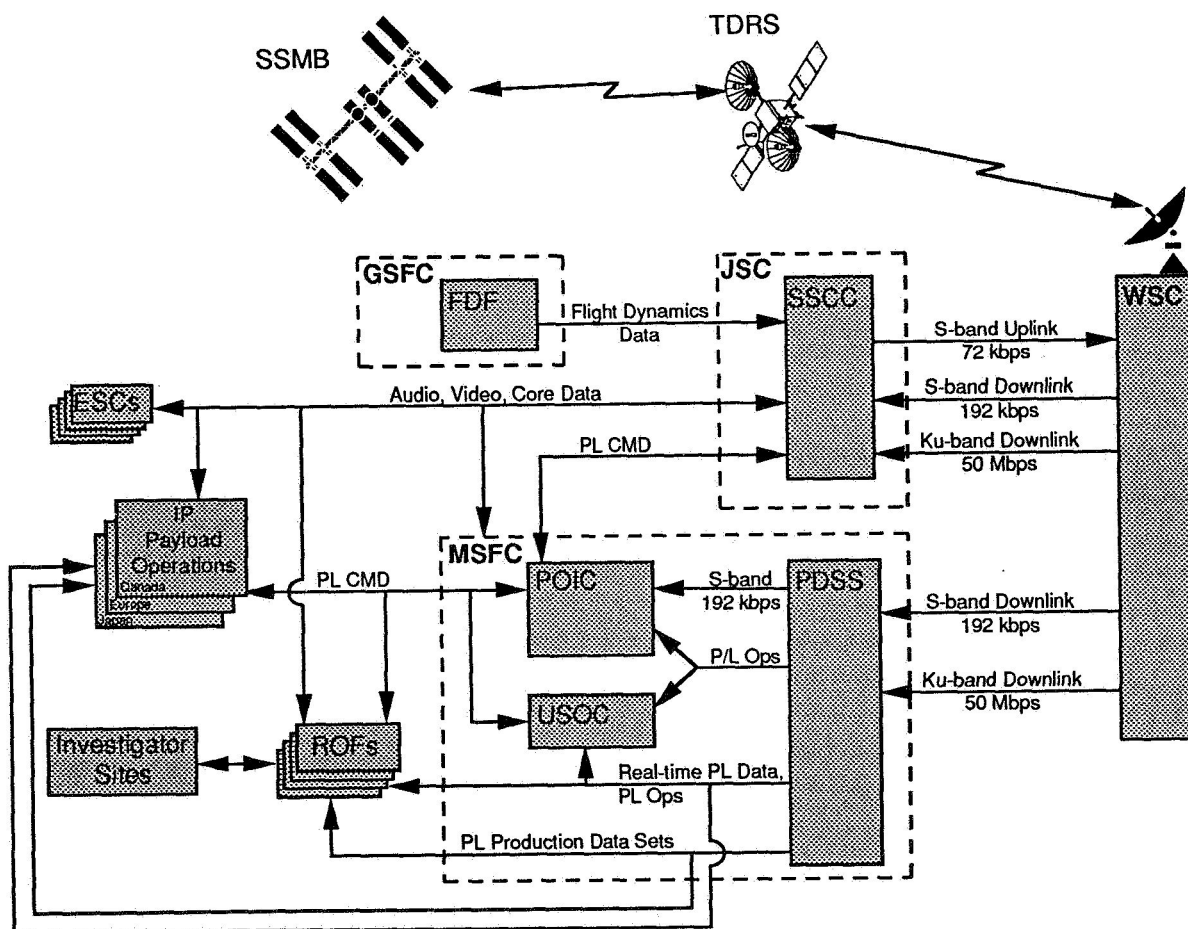
- Management and integration of payload operations and payload operations planning
- Interface to the users for audio, payload commanding, and payload file transfer
- Integration and transmission of payload command streams to the SSCC.

FDF The Flight Dynamics Facility, located at Goddard Space Flight Center (GSFC) in Greenbelt, Maryland, will provide orbit and attitude data to support TDRSS acquisition and SSMB position determination.

PDSS The Payload Data Services System, located at MSFC, will be responsible for payload data distribution. The PDSS will perform the following functions:

- S- and Ku-band data capture
- Production data processing and archiving
- Real-time and processed data distribution
- Line outage recording and data rate buffering.

ESC Geographically dispersed Engineering Support Centers will provide maintenance and sustaining engineering support in various areas of SSMB systems expertise.



USOC NASA will provide the U.S. Operations Center at MSFC to house the U.S. payload operations capability. In addition, U.S. users may establish Remote Operational Facilities (ROFs) at geographically dispersed sites.

IP Interfaces Each International Partner (IP) will access the SSF ground data system at a dedicated interface point located within the continental United States. IPs are responsible for data transport between the interface points to their various operational centers, which are described in Section 6.

4. CORE OPERATIONS

Core operations are discussed in terms of the uplink and downlink data paths.

4.1 Core Data Uplink Path

SSCC operators will initiate core commands. Core commands and data files will then be generated at the SSCC, merged with the POIC uplink data stream, multiplexed with multiple voice channels, and transmitted to the WSC for uplink to the SSMB via S-band at 72 kilobits per second (kbps).

4.2 Core Data Downlink Path

The 192 kbps S-band downlink will be formatted in accordance with the Consultative Committee for Space Data Systems (CCSDS) recommendations for Advanced Orbiting Systems (AOS). The data stream will consist of CCSDS packets containing multiple voice channels, payload safety telemetry, and core telemetry.

Core telemetry will be downlinked via TDRSS and forwarded to the SSCC. The SSCC will process and archive these data for use by SSCC operators and ESC engineers. Payload users will also have access to archived ancillary parameters, but will probably prefer to build their own customized ancillary data sets on board and have them downlinked and routed to the PDSS.

Core video data and core data originating from the on-board recorders will also be archived at the SSCC. Recorder data will be archived with all other core data, but the manner in which

these data packets will be time ordered with other data for later retrieval is undetermined. The optimal method of video archiving is also still under study.

IP core data will be routed to the locations described in Section 6.

4.3 Core Operations Security

Consistent with the requirements to protect the health and safety of manned missions, the SSFP uplink command data stream will be encrypted at the SSCC and decrypted on board.

5. PAYLOAD OPERATIONS

Payload operations are discussed in terms of the uplink and downlink data paths.

5.1 Payload Uplink Data Path

Payload users will generate commands and data files for uplink to their respective payloads. These data may be transmitted in real time or non-real time to the POIC. At the POIC, the various payload command loads will be combined into an integrated payload command load and transmitted to the SSCC for uplink.

IP payload commands and data files will be received by the POIC from the locations described in Section 6.

5.2 Payload Downlink Data Path

The Ku-band downlink will be formatted in accordance with the CCSDS AOS recommendations. The data stream will consist of 13 Virtual Channels (VCs) at a composite data rate of 50 megabits per second (Mbps). Of these VCs, 7 will carry high-rate payload data, 1 will carry low-rate payload data, 4 will carry SSF video, and 1 will contain fill data sufficient to ensure a constant downlink data rate.

The data stream will be downlinked via TDRSS to the WSC and subsequently broadcast via domestic communications satellite to both the SSCC and the PDSS.

The PDSS will capture and store the 50 Mbps broadcast from WSC for a limited time to protect against possible line outage or system failure. The PDSS will demultiplex the captured data in accordance with the CCSDS AOS formats.

The PDSS will separate the VCs and route the seven high-rate streams according to Virtual Channel Identifier (VCID). However, as the low-rate VC consists of data from multiple payloads, it must also be demultiplexed at the CCSDS packet level.

The PDSS will distribute selected VC and packet data streams to users in real time. These data will also be archived off-line for a period of two years. Archival will occur at the VC level. No further organization or labeling of the data will occur; consequently data requestors must know the channel and time at which the data were originally time-stamped.

The PDSS will also perform Level-Zero Processing (LZP) on selected VC data streams. LZP includes the removal of communications process artifacts such as headers and fill data, removal of data redundancies, time-ordering of data packets, and generation of data quality reports. LZP data will be archived on-line at the PDSS for a period of four days. Users may request and receive this data within 24 hours. The request must include the time of data production as well as the destination identification.

The PDSS will also provide "rate buffering" of selected data to match user ingest capabilities.

In addition, the PDSS will capture and demultiplex the 192 Kbps S-band data stream and forward it to the POIC in real-time to support payload operations.

Payload video from the SSF video system (not to be confused with payload digital image data, which is downlinked within the payload data stream) will be routed to JSC for distribution as a National Television Standards Committee (NTSC) analog signal.

Payload data, audio, and video will be routed to the IP locations described in Section 6.

5.3 Payload Operations Security

Payload downlink data will be protected to various degrees as they traverse the data system. Users who require additional protection (e.g., for sensitive corporate data or data whose release would be inconsistent with national policies on technology transfer) may employ their own protection mechanisms, such as encryption.

6. INTERNATIONAL PARTNERS

The IPs will develop and staff operational centers to support their payloads and modules. These centers will be compatible with the POIC. Data will be routed to and from the centers via gateways located within the continental United States.

CSA The Canadian Space Agency will provide the Payload Operations Center (POC) in Canada to support its robot arm and payloads.

ESA The European Space Agency will provide the European Payload Operations Control Center (EPOCC) at Oberpfaffenhofen, Federal Republic of Germany, to support the Columbus Attached Pressurized Module and European payloads.

NASDA The Japanese National Space Development Agency will provide the Space Station Integration and Promotion Center (SSIPC) located at Tsukuba Space Center (TKSC) near Tokyo, Japan. SSIPC will support the Japanese Experiment Module (JEM) and Japanese payloads.

7. FUTURE CHALLENGES

As the SSFP evolves over its 30-year life span, the ground data system will face significant challenges. As more numerous and complex payloads are added to the SSMB, downlink data rates are projected to climb from 50 Mbps to as much as 300 Mbps. With the addition of a Ka-band communications capability, data rates could exceed 300 Mbps. Global Positioning System (GPS) receivers may be added to the SSMB to enhance tracking capabilities. In addition, the payload users are expected to become far more geographically

dispersed. The network that supports the SSF must therefore be modular, easily expandable, and robust.

The SSFP will face these and other unforeseen requirements as they evolve. But the planned ground data distribution system will provide a solid basis for this growth and will allow NASA and its International Partners to carry out manned, scientific spaceflight well into the next millenium.

EARTH OBSERVATION ARCHIVE ACTIVITIES AT DRA FARNBOROUGH

N94-23958

M D Palmer¹ and J M Williams²

1. Data Technology Section, Remote Sensing Department, Space Sector, DRA (Aerospace),
Farnborough, Hampshire, UK, GU14 6TD

2. Serco Space (Consultancy Division), Southall, Middlesex, UK, UB2 5NJ

ABSTRACT

Space Sector, Defence Research Agency (DRA), Farnborough have been actively involved in the acquisition and processing of Earth Observation data for over 15 years. During that time an archive of over 20 000 items has been built up.

This paper describes the major archive activities, including:-

- operation and maintenance of the main DRA Archive.
- the development of a prototype Optical Disc Archive System (ODAS).
- the catalogue systems in use at DRA.
- the UK Processing and Archive Facility for ERS-1 data.
- future plans for archiving activities.

Key Words:- Data archiving, earth observation, catalogue systems.

1. INTRODUCTION

Space Sector, Defence Research Agency, Farnborough (formerly RAE Farnborough) have been actively involved in the acquisition and processing of Earth Observation data for over 15 years. During that time an archive of over 20 000 items has been built up. Serco Space has had a long

association with the operations of Space Sector and is currently involved with a number of activities concerned with improving the efficiency of archive operations.

This paper provides a summary of the major archive activities under the following headings:-

- maintenance of the DRA Archive.
- Optical Disc Archive.
- Catalogue Systems.
- ERS-1 Archive.
- Future activities.

2. MAINTENANCE OF DRA ARCHIVE

Tapes in the main DRA archive are divided into three categories:-

- Master - the original CCT as received from the satellite groundstation (such as the DRA groundstation at Lasham) or data distributor.
- Copy - a backup copy of the master tape is held in a separate building.
- Image - a transcription of the original tape into a standard format (RAE format).

At the time of writing (October 1992) the DRA Archive contained the image sets shown in Figure 1.

CATEGORY	NUMBER OF IMAGES
Landsat (TM and MSS)	2070
SPOT	782
Geosynchronous (Primarily Meteosat)	142
Seasat	1826
Geocoded Images (Landsat & SPOT)	179
Various Tapes	8954
AVHRR	2578
Total	16531

Figure 1 - Breakdown of DRA Archive.

Note that the number of images is less than the total number of tapes in the archive for two reasons:-

- large images (such as Landsat-TM) occupy more than one tape.
- a single listed image can occur as multiple format tapes (eg, master, copy, image).

The 'various tapes' category includes system back-ups, processed images, airborne SAR and thematic mapper campaigns, and DRA products such as the AVHRR mosaics of Europe and Antarctica.

All tapes are maintained in a temperature and humidity controlled environment (see Figure 2):-

ENVIRONMENTAL ASPECT	VALUE
Temperature	18°C to 24°C
Rate of change of temperature	Less than 6°C/hour
Relative Humidity	40% to 60%

Figure 2 - Recommended Storage Conditions for Open Spool Tapes (based on BS4783: Part 2)

Archiving activities include issue and recall of copy tapes, tape maintenance and evaluation and recycling of 'scratch' tapes.

3. OPTICAL DISC ARCHIVE SYSTEM

In 1985 the DRA commenced a programme with the objective of transferring the important elements of the

archive to optical disk. The plan for the three stages was as follows:

- procure hardware and develop software for controlling optical disc drives.
- implement a semi-operational prototype system
- implement an interoperable catalogue system to control the optical disk archive

Stage 1 of the program was completed, by Logica Space and Defence Systems Limited, in 1988 and stage 2 in 1990. Stage 3 has not currently been initiated due to changing requirements within DRA.

ODAS can perform online archive functions such as accessing optical discs to read or write data, as well as catalogue functions such as searching a database.

The ODAS hardware consists of an Integrated Automation jukebox with ATG Gigadisc drives interfaced to a PRIME 2655 computer.

4. CATALOGUE SYSTEMS

To support the smooth operation of the Archive, efficient cataloguing systems are essential. Catalogue systems currently in use at DRA include:

- REMIS.
- Photolab Management and Administration System (Photo-MAS).
- PARADOX Tape Catalogue System.

4.1 REMIS

REMIS is the main catalogue system for the DRA Archive. It is hosted on a PRIME computer and based on the INFO database system. It is used by the Tape Librarians to enter and validate holdings and by a variety of users to perform data searches.

REMIS uses two indexes to maintain uniqueness of records:-

- tape identification numbers
- physical location code (building, rack, slot).

Three main types of search can be performed:-

- retrieve information corresponding to a known tape number.
- composite query for a specific image category (eg, SPOT).
- area search (lat, long) across all image categories based on nominal scene centres.

The results of a typical REMIS search can be seen in Figure 3. The constraints specified were (Satellite = LANDSAT) AND (Sensor = TM) AND (Date ≥ 1-9-86) AND (Date ≤ 30-9-86).

19/10/1992

PAGE 1

LANDSAT RETRIEVAL

SATPATH/ROW	QUAD	DATE	SEN	CC	LATITUDE	LONGITUDE	COUNTRY/AREA
5 196 27.0	4	6/9/86	TM	1213	46.70N	6.63E	FRANCE/SWITZERL
AND							
5 204 21.0	3	14/9/86	TM	1111	55.42N	2.73W	UK
5 204 21.0	1	14/9/86	TM	3000	56.84N	2.49W	UK
5 204 22.0	0	30/9/86	TM	7173	54.52N	2.96W	UK

Key:

SAT	Satellite Number.
QUAD	Image Quadrant (0 = whole image).
CC	Cloud Cover (0 - 8 for NW, NE, SE, SW of image).

Figure 3 - Results of REMIS Search

4.2 Photolab Management and Administration System (Photo-MAS)

The Photo-MAS was developed to enable records of Archive resources used by the Space Sector Photolab to be maintained. The Photo-MAS maintains a catalogue of film and print holdings as well as an 'ancestry record' showing the original tapes that the product was derived from.

The overall Photo-MAS system can be seen in Figure 4. Information stored in the MAS database is transferred, via the MAS executive to Photolab operators, and onward to customers. The modular design of the Photo-MAS will enable a direct link to certain customers to be provided at a later stage, if required.

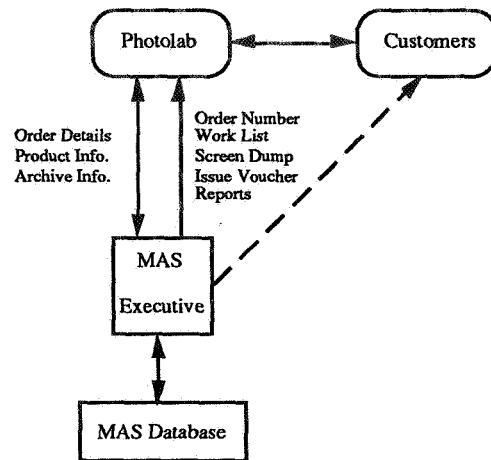


Figure 4 - MAS Top-Level Model

The system runs on a SUN-4 WorkStation using the ORACLE RDBMS.

4.3 PARADOX Tape Catalogue System

The PARADOX Tape Catalogue System runs on a standard IBM-PC 386 compatible computer with software developed using the PARADOX Application Language (PAL).

The system is based around three databases:

- image types database, which contains unique records for specifying each category of image
- storage location database specifying all possible physical locations of storage media
- image catalogue database which contains the main database records

The user can perform a variety of operations including a number of search options. These include point searches (to determine whether a point lies within an image boundary) and area searches (to determine the percentage overlap between an image and a specified area).

Distributed users of the system can be supplied with a run-time version of the system and regular updates of the database on 5 1/4" or 3 1/2" floppy disk.

5. ERS-1 ARCHIVE

DRA Farnborough is also involved with the European ERS-1 mission. Serco Space is responsible (under subcontract to National Remote Sensing Centre Limited) for operating the Earth Observation Data Centre (EODC) at Farnborough which constitutes the UK Processing and Archive Facility (UK-PAF). Data is stored in a purpose-built archive designed to hold HDDTs and Optical Discs.

The major activity of the UK-PAF is the archiving and processing of Synthetic Aperture Radar (SAR) data from ERS-1. The unprocessed data is stored on HDDT with the resultant format on CCT. The Centre also archives low-bit rate data (primarily Radar Altimeter and Scatterometer data), with raw data incoming on optical disc and products stored on Exabyte cartridge. At the time of writing (October 1992) the archive contained approximately 5000 items (see Figure 5 for details). Eventually the archive will store 10 000 HDDTs representing two years worth of operational SAR data.

MEDIUM	DATA TYPE	NUMBER
Optical Disc	Low-Bit Rate (LBR) Data	1688
HDDT	SAR Data (Unprocessed)	3184
CCTs	SAR Products	100
Exabyte Cartridge	LBR Products	200
Total		5172

Figure 5 - Contents of UK-PAF Archive.

Every tape booked in is subjected to Quality Control (QC) processing before an archive report is generated. The report is then despatched to ESA to be stored on their Central User Services (CUS) database.

6. FUTURE ACTIVITIES

Future archiving activities at DRA Farnborough are in an early stage of planning but possible directions include:-

- preparation for involvement in the archiving activities of future European Earth Observation Missions (eg, ERS-2, POEM-1).

- archiving of digital cartographic and other GIS data.
- transfer of data on magnetic media to optical disk.
- general adoption of cheaper, more flexible catalogue systems using standard components.
- improved QA and configuration control.
- assessing catalogue inter-operability requirements.

7. ACKNOWLEDGEMENTS

Thanks are due to the following staff members of DRA Space Sector and Serco Space for their assistance in compiling this paper:- Barry Marsh, Linda Henderson, Ted Wood, Dr Graham Atkin.

REFERENCES

- 1 British Standards Institution,
British Standard: Storage, transformation and maintenance of magnetic media for use in data processing and information storage,
BS 4783 (4 Parts) 1988.
- © British Crown Copyright 1992 / DRA
Published with the permission of the Controller of Her Britannic Majesty's Stationery Office.

SESSION L

OPERATIONS MANAGEMENT

CO-CHAIRPERSONS

J. E. Leibee, NASA Goddard Space Flight Center, USA
G. P. Textor, JPL, California Institute of Technology, USA

Summary	843
G. P. Textor	
L.1 Operational Characterisation of Requirements and Early Validation Environment for High-Demanding Space Systems	845
E. Barro, A. Del Bufalo, and F. Rossi, Compagnia Italiana Servizi Tecnici (CISST) S.p.A., Rome, Italy	
L.2 Toward Lowering the Cost of Mission Operations	851
S. D. Wall, JPL, California Institute of Technology, Pasadena, California, USA; K. W. Ledbetter, NASA, Washington, D.C., USA	
L.3 New Mission Requirements Methodologies for Services Provided by the Office of Space Communications	857
D. P. Holmes and J. R. Hall, JPL, California Institute of Technology, Pasadena, California, USA; W. Macoughtry, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA; R. Spearing, Telos Corporation, Chantilly, Virginia, USA	
L.4 From Requirements to Acceptance Tests	863
L. Baize and H. Pasquier, CNES Toulouse Space Centre, France	
L.5 Resource Allocation Planning with International Components	867
G. Burke, R. Durham, and F. Leppla, JPL, California Institute of Technology, Pasadena, California, USA; D. Porter, California Institute of Technology, Pasadena, California, USA	
L.6 An Intelligent Planning and Scheduling System for the HST Servicing Missions	875
J. Johnson, L. Bogovich, A. Tuchman, A. Kispert, B. Page, C. Burkhardt, R. Littlefield, and D. McLean, Bendix Field Engineering Corporation, Seabrook, Maryland, USA; W. Potter, W. Ochs, and A. Dougherty, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA	
L.7 Scheduling the Future NASA Space Network: Experiences with a Flexible Scheduling Prototype	881
N. Happell and J. Minnix, Stanford Telecommunications, Inc., Reston, Virginia, USA; K. L. Moe, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA	
L.8 Software Reuse in Spacecraft Planning and Scheduling Systems	887
D. McLean, A. Tuchman, T. Broseghini, W. Yen, B. Page, J. Johnson, L. Bogovich, C. Burkhardt, J. McIntyre, S. Klein, H. Dao, and R. Littlefield, Bendix Field Engineering Corporation, Seabrook, Maryland, USA; W. Potter, NASA Goddard Space Flight Center, Greenbelt, Maryland, USA	



L.9 Planning: Management of Predictability and Uncertainty and Keeping Abreast of Developments	893
C. Bastien-Thiry, CNES Toulouse Space Centre, France; G. Verfaillie, Centre d'Études et de Recherches de Toulouse, France	

L.10 Estimating the Deep Space Network Modification Costs to Prepare for Future Space Missions by Using Major Cost Drivers	899
D. S. Remer, J. Sherif, and H. R. Buchanan, JPL, California Institute of Technology, Pasadena, California, USA	

SUMMARY —SESSION L

G. P. TEXTOR

Jet Propulsion Laboratory

First, I would like to thank my co-chair, Jack Leibee, who helped in selecting and scheduling the presentations but unfortunately was ill on the day of the session. I also want to thank Tom Ryan, who volunteered to fill in for Jack on the day of the presentations and greatly assisted in the session.

The session was well attended and finished on time. Five themes were discussed:

- Lowering operations cost
- Scheduling processes
- Planning, scheduling, and testing using intelligent tools
- Requirements tracing
- Cost estimating

The papers' presenters and synopsis of topics are provided below. (The paper L.1, "Operational Characterisation of Requirements and Early Validation Environment for High-Demanding Space Systems," was not presented.)

L.2, "Toward Lowering the Cost of Mission Operations," S. D. Wall. This presenter discussed cost drivers and ways in which costs of mission operation systems may be contained. He emphasized that early attention to concept and requirements definition and improved communications between software developers, testers, and users can reduce development costs.

L.3, "New Mission Requirements Methodologies for Services Provided by the Office of Space Communications," D. P. Holmes. This paper was a discussion of processes, tools, and the impact on customers of space communications services on mission planning and operations. The goal is to minimize overall flight/ground costs by using the Mission Requirements Request (MRR) document as a living document in the Phase A studies and throughout the life of the project. This will make it possible to fit mission requests with Deep Space Network capabilities.

L.4, "From Requirements to Acceptance Tests," L. Baize. This presentation described concepts of tracing a development from the definition of the

user requirements to the accepted software system using database software on a Macintosh.

L.5, "Resource Allocation Planning with International Components," G. Burke. A method and challenges of scheduling international components were discussed. The necessary first step is to integrate and standardize planning databases in order to identify potential cross-support. However, a decentralized communications process and a wider variety of automated planning tools will be required.

L.6, "An Intelligent Planning and Scheduling System for the HST Servicing Missions," L. Bogovich. This presentation described how the Service Mission Planning and Replanning Tool is able to help Hubble Space Telescope personnel more efficiently build timelines and command plans that are required to support the servicing missions.

L.7, "Scheduling the Future NASA Space Network: Experiences with a Flexible Scheduling Prototype," K. L. Moe. This presenter discussed a concept of flexible scheduling for the Space Network Control System by utilizing the Space Network User POCC Interface (SNUPI) at each Project Operations Control Center (POCC).

L.8, "Software Reuse in Spacecraft Planning and Scheduling Systems," D. McLean. This was a discussion of efforts to reuse C-based software in spacecraft planning and scheduling systems.

L.9, "Planning: Management of Predictability and Uncertainty and Keeping Abreast of Developments," C. Bastien-Thiry. This was a discussion of tools to help in drawing up and testing a space mission plan, such as for the Hermes spaceplane.

L.10, "Estimating the Deep Space Network Modification Costs to Prepare for Future Space Missions by Using Major Cost Drivers," D. S. Remer. This presenter discussed a model to do long-range planning cost estimates for the Deep Space Network. With actual data from three missions, the model was used to predict the cost of two missions and was accurate to better than 20%.

N94-23959

OPERATIONAL CHARACTERISATION OF REQUIREMENTS AND EARLY VALIDATION ENVIRONMENT FOR HIGH DEMANDING SPACE SYSTEMS

E. Barro, A. Del Bufalo, F. Rossi

CISSET S.p.A.
Rome, Italy

ABSTRACT

The definition of some modern high demanding space systems requires a different approach to system definition and design from that adopted for traditional missions. System functionality is strongly coupled to the operational analysis, aimed at characterising the dynamic interactions of the flight element with its surrounding environment and its ground control segment. Unambiguous functional, operational and performance requirements are to be defined for the system, thus improving also the successive development stages. This paper proposes a Petri Nets based methodology and two related prototype applications (to ARISTOTELES orbit control and to Hermes telemetry generation) for the operational analysis of space systems through the dynamic modelling of their functions and a related computer aided environment (ISIDE) able to make the dynamic model work, thus enabling an early validation of the system functional representation, and to provide a structured system requirements data base, which is the shared knowledge base interconnecting static and dynamic applications, fully traceable with the models and interfaceable with the external world.

Key Words: System Operations, Functional Analysis, Dynamic Simulation, Validation, Petri Nets.

1. OPERATIONAL FEATURES OF HIGH DEMANDING SPACE MISSIONS

Some of the planned European space missions (Low Earth Orbit or Deep Space missions, like ARISTOTELES, Cassini, Mars mission) present a significant increase of complexity in the spacecraft system definition with respect to the traditional communications or scientific satellites.

The main reason is that their specific operational constraints (short and rare ground contact periods, or decreasing ground control capabilities and performances due to long distances) have for this kind of missions a stronger impact on system architecture.

Such constraints impose the need to define a high degree of on board autonomy for the spacecraft, or, in other words, to identify specific operations driven flight element control functions which in the

traditional missions were typically allocated to the ground segment, finding a good compromise between cost and complexity of a self standing system and the operational risks associated with the delegation of tasks.

In addition, for these missions the system operator can rely on a limited budget of information about the spacecraft, which must be carefully defined in order to ensure the safety of the spacecraft and to optimise both the system monitor and control loop and the payload exploitation.

Such kind of problems can never be solved only on the basis of a previously consolidated experience in "similar" past missions, as the definition of system autonomy and the consequent spacecraft design are heavily constrained by the dynamic interactions of the flight element with its surrounding environment and its ground control segment, which are strictly mission specific.

2. AN ALTERNATIVE APPROACH TO SYSTEM DEFINITION AND OPERATIONS DESIGN

As a matter of fact, the correctness in the identification of the optimum sharing of functions between on board and ground is critical for the definition of suitable functional and performance requirements for the spacecraft, which are the baseline for the system architecture.

As a consequence, an in depth analysis of the operations related aspects of modern spacecrafts (and therefore of the system dynamic behaviour) is fundamental even in the early system functional analysis and requirement specification phase of the spacecraft development.

Due to the relevance of the spacecraft operational aspects in the definition of the system architecture, it is very important in this phase to demonstrate that the system built by the designer is capable of fulfil in its working environment the identified functionality, especially if the system functioning is subject to severe time constraints.

The current analysis methodologies used for the system definition phase take only partially into account the operational aspects, and ensure only a preliminary coherence of the system functional model with the derived requirements.

In this phase usually the designer:

- o establishes some operational choices for the system, taking into account the constraints imposed by the context;
- o identifies a hierarchical structure of system functions, together with their relevant attributes;
- o formalises the system functions into a set of functional and performance requirements;
- o builds the first architectural design of the system, where the system requirements are translated into a physical architecture, on the basis of the a priori implementation constraints imposed by the user.

The functional model, however, is not currently able to provide an exhaustive representation of the system, as the usual modelling methodologies (e.g. SADT, OODLE) are all static.

The system dynamics, i.e. the representation of its dynamic behaviour and the modelling of the system operations is generally not taken into account in the system definition phase.

As a consequence the specification of dynamic requirements for the system is not usually derived from the characteristics of the model.

This lack in the system definition suggests the need to introduce a more complete and consistent approach to this phase, in order to tightly link the flight element functions and the related static and dynamic requirements to the operations concept identified for the spacecraft, ensuring their full consistency.

This approach can be enforced by exploiting an support environment aimed at providing the system designer with aids for the generation of a complete and fully traceable model of the system at functional level, by means of:

- o modelling the system static and dynamic behaviour;
- o building a coherent and consistent set of system requirements;
- o validating the model versus the system requirements and the operational strategies;
- o providing a significant control over the next steps of system life cycle (system architectural design).

Such a support environment, named Integral System Investigation and Definition Environment (ISIDE), is currently being developed by Ciset; its basic principles and features are described in section 5.

3. A TYPICAL EXAMPLE: THE ARISTOTELES MISSION

A subset prototype of ISIDE, the System Dynamic Analysis Environment, has been developed and successfully used in the field of spacecraft operations analysis in the frame of ARISTOTELES phase pre-B

studies, related to satellite autonomy concept definition (Ref. 2, 3).

Such a prototype provides the capability to build and execute a dynamic executable functional representation of the system (based on the well known Petri Nets methodology, Ref. 1) or of an operational process.

The representation is parametrised by means of a direct link with a System Requirements Data Base.

The objective is to verify the validity of operations concepts with respect to system requirements and to the spacecraft operational context.

The representation (model) can then be run as a real simulator, providing as an output statistical figures of the system dynamics and enabling the assessment of the correctness of the tested operational approach and the identification of critical paths in the process execution (bottlenecks, deadlocks).

The basic steps of model generation are:

- o the building and parametrisation of a dynamic functional representation of the process (built with Petri Nets methodology): the representation is obtained from a static SADT model by means of translation rules and by adding to the static model the system dynamic information (missing in SADT) as derived from the Requirements DB.
- o the execution of an ad hoc simulation, the output of which enables the validation of the concept under study and the generation of statistical information on the model behaviour.

3.1 Description of the model

The prototype has been extensively used for the analysis and early validation of a proposed operational concept of ARISTOTELES, a very low earth orbit satellite ($h=200$ Km, $i=98.5^\circ$), hosting a gravity gradiometer aimed at an accurate measurement of the earth gravity field.

The strategy for keeping the satellite inside its allowed deadband (± 3 Km), through the execution of dedicated Orbit Raise Manoeuvres (ORM), was critical due to reduced GS coverage (only Kiruna ground station available), limited S/C weight and high decay rate (about .240 m/h)

Further constraints came from the S/C critical safety conditions and the limited accuracy of orbit determination process during specific contingency situations.

The autonomy concept definition, therefore, focused on the splitting of orbit control process functions between the ground and the space segment.

The basic choice was thus represented by identifying where to perform the computation of the predicted times and amount of orbit raise manoeuvres, combining it with a suitable operational strategy and with the constraints coming from the environment.

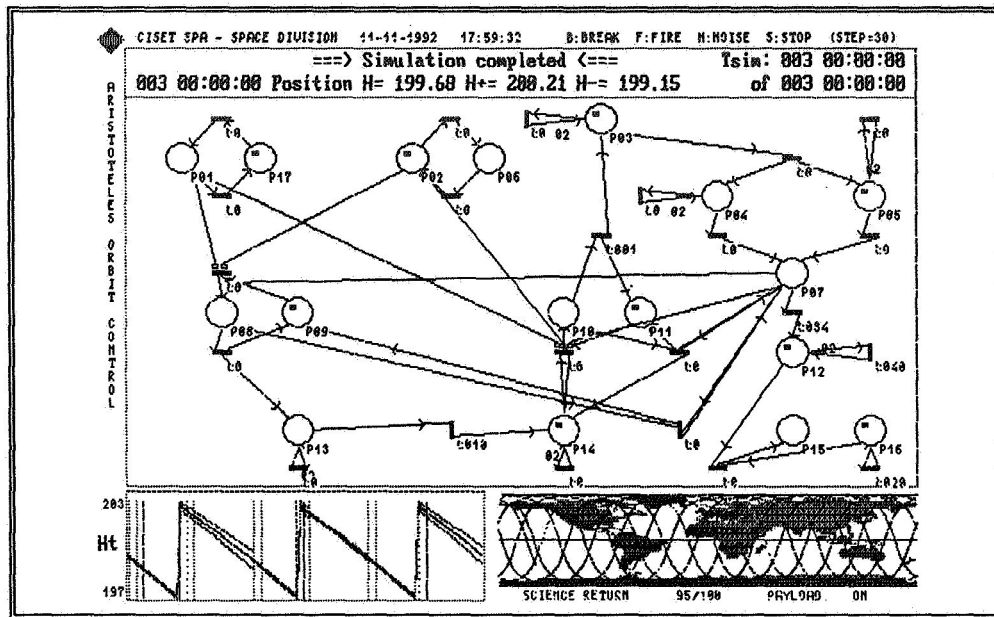


Fig. 1: ARISTOTELES Orbit Raise Manoeuvre Model simulation output.

A dynamic model of the ORM was built and put at work. In figure 1 the final graphical output of a 3 days model simulation is presented. The following table shows the significance of the network places.

1	Ground Contact enabled (no failure)
2	Kiruna is visible
3	Next Manoeuvres times on board
4	Man. n ready for execution
5	Man. n+1 ready for execution
6	Kiruna is not visible
7	Manoeuvre in execution
8	Down link in execution
9	Down link standby
10	Up link in execution
11	Up link standby
12	Manoeuvre successfully executed
13	Satellite Tracking executed
14	Next Man. times computed on ground
15	Satellite out of deadband
16	Satellite within the deadband
17	Ground Contact inhibited (failure)

Table 1: ORM Petri Net places description.

The model is based on a Petri Nets 'engine' describing the overall functional mechanism of the process, including ground functions (tracking, manoeuvres times computation and up-link), spacecraft functions (manoeuvres execution, down-link) and the spacecraft environment influence on the process (ground station visibility, contact failures), which schedules the various simulation modules.

Two major external simulators are interfaced:

- o an orbital propagator, driving the spacecraft visibility on the basis of its initial position, orbital parameters and of Kiruna features;
- o an atmospheric drag model, which computes the satellite altitude (including altitude determination

errors) on the basis of a drag simulation algorithm, taking as an input from the network the manoeuvres executed and releasing as an output the current satellite altitude.

The Petri network is parametrised with the dynamic information about the process (e.g. altitude deadband, characteristic times) which are derived a priori from a database of system requirements. The graphical display, as shown, combines the Net with the output of the two simulators (on the bottom).

3.2 Simulation results

The simulation of ORM process for different initial conditions and environmental conditions enabled the validation of the tested operations strategy, once fixed the value of system parameters provided within the system requirements database.

Furthermore it allowed to verify the sensitivity of the strategy to the variation of any of the parameters of the model.

Finally, the simulation execution provided a wide number of statistical results about the process under study, like the distribution of manoeuvres intervals and of manoeuvres size, the deadband utilisation figure, the scientific return comparing those values with the expectations at System Requirements level.

4. CHARACTERISTICS OF THE PROTOTYPE MODELLING ENVIRONMENT

The System Dynamic Analysis Environment used for ARISTOTELES ORM model was developed on IBM PS2 using C language under DOS 5.0.

The prototype architecture, as shown in figure 2, assembles three separate environments:

- o a system modelling environment (PN, SRDB, simulation modules and link editors);
- o a simulation execution environment;
- o an evaluation environment.

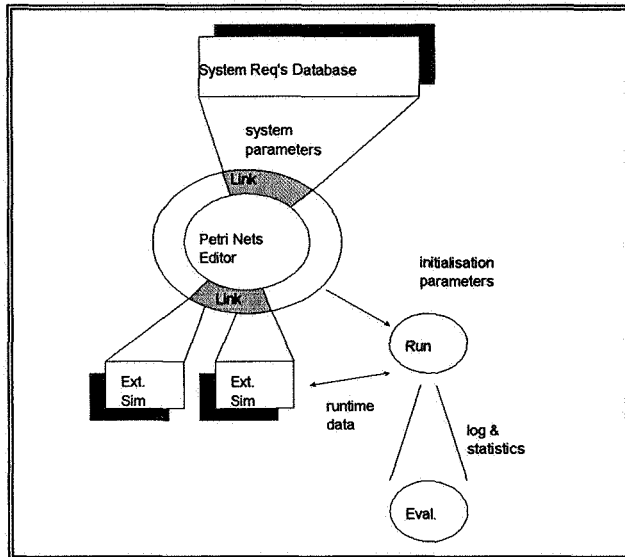


Fig. 2: Modelling environment architecture.

The model preparation is based on a Petri Nets Editor with the following characteristics associated with the network transitions:

- o multiple and inhibitor arcs;
- o deterministic firing time;
- o firing conditions (including random);
- o actions executed on transitions firing (e.g. activation of simulation modules).

4.1 Model preparation and system requirements

By means of this editor the system engineer can build the network which models the process (or the system) under study, defining the process mechanism and the related transition characteristics (firing time, conditions, actions), and identifying the set of data, variables or commands which constitute the interface of the model with any external software (e.g. an external simulator).

The editor also enables the creation of a link of variables with a **System Requirements Database**, which can be generated and maintained separately by means of a database editor.

Whenever the database information is changed, the network parameters used for the simulation run are updated accordingly.

4.2 Simulation Execution

Once the model has been generated, a simulation can be executed by means of the run-time module.

All the model parameters derived from the system requirements database can be accepted or modified in this phase. In addition, other simulation initialisation parameters, like simulation time step can be set.

The run-time module executes the simulation according to the Petri Nets syntax, invoking external simulation modules for conditions verification and actions performing. The capability of defining firing conditions for the network transitions enables the implementation of functional priorities, in case the modelled process is fully deterministic (no resource conflict between concurrent functions is allowed).

The definition of transitions associated actions enables the parametrisation of network tokens, modelling in this way the availability of different kind of resources within the system.

All the significant simulation events and parameters (transitions firing, parameters values) are displayed and logged. The display messages can be defined in a customised way during the model preparation, and may include the monitor current values of model internal and external variables.

4.3 Simulation Evaluation

After the simulation execution, the log file is processed by an **Evaluation** module, which computes and displays the main network statistics, i.e. for each transition:

- o overall number of firings;
- o minimum, average and maximum time between two successive firings.
- o pre-defined statistical figures of selected network parameters.

The module also allows the navigation within the log file (e.g. searching for all the occurrence of a pre-defined event).

5. INTEGRAL SYSTEM IDENTIFICATION AND DEFINITION ENVIRONMENT (ISIDE)

The above described System Dynamic Analysis Environment is a preliminary application of a more general concept, ISIDE.

ISIDE is aimed at providing a computer aided environment for the generation of an integral and consistent system description and for its validation in the frame of the system definition phase.

The fundamental idea behind ISIDE is the integration of a functional static and dynamic representation of the system and its specification into a set of system requirements, addressing an integral system model, where all the system related information are coherently collected.

From the ISIDE viewpoint the system requirements have not to be considered as a further information of

the system, but they grow up together with the functional and dynamic models, being strictly linked to them by means of the ISIDE syntax.

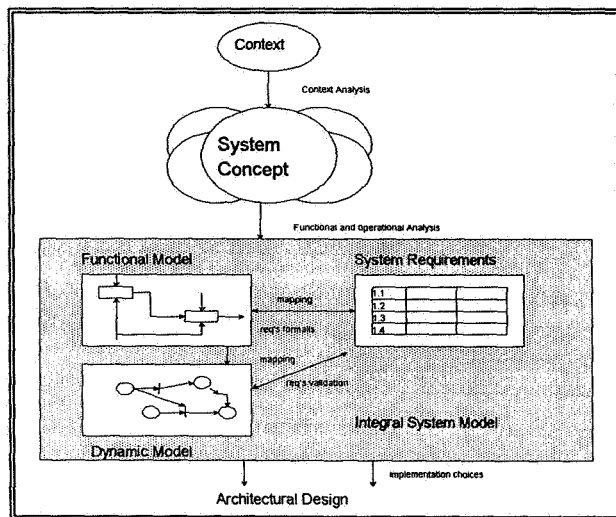


Fig. 3: ISIDE Concept.

The models enable the early verification of the correctness of the system concept, and as a consequence of the fitness of the model to the user needs. In addition, the possibility of describing the system dynamics enables even in the architectural design phase the verification at a functional level of the choices made, in terms of system functioning.

5.1 Summary of ISIDE Features

ISIDE will provide the following capabilities:

- 1) Modelling the system static and dynamic behaviour.

ISIDE defines a rigorous syntactic link between the static and dynamic models, in order to ensure they describe exactly the same system (the dynamic model is "automatically" derived from the functional model).

- 2) Building a coherent and consistent set of system requirements.

The system requirements are considered in the context of ISIDE the core of the system representation. They are structured, parametrised and directly interfaced with the entities and parameters of the models.

This ensures traceability with the models, providing full flexibility of the system representation.

- 3) Validating the model.

The dynamic model provides the capability to show, via an executable simulation, that the system works, at functional level, in compliance with user needs, time constraints and operational choices reflected in the system concept.

- 4) Controlling system life cycle.

The representation capability of the dynamic model can also be exploited in the next phases of system life cycle, where the model can be easily enriched with additional parameters coming from the implementation choices.

From the point of view of ISIDE implementation, the environment is built by means of the integration of:

- o an SADT functional modelling tool;
- o a Petri Net dynamic modelling tool;
- o an ORACLE based system requirements DB.

The use of ORACLE and C based interfaces ensures ISIDE will be fully open to the external world, thus enabling a wide utilisation of ISIDE (e.g. integration with external simulators, use of the database along different phases of the system life-cycle) as specific functional or data interface may be set for the exchange of relevant parameters.

On the other hand, the system knowledge base may also be easily maintained, evolving in the various phases of system definition, up to becoming a real operational database of the system.

5.2 Application of ISIDE to Architectural Design

As already outlined, although the ISIDE concept was born for covering lacks in the early system definition phase, the environment characteristics make it effective to exploit ISIDE also during the successive phases of the system life cycle, and in particular during the architectural design phase. That is natural when considering that the hierarchical nature of the methodologies used for system description (SADT and Petri Nets) enables the progressive detailing of the model in parallel with the system evolution.

As an example of an application of ISIDE to system design we propose a model, generated and executed using the System Dynamic Analysis Environment, of a software for the simulation of Hermes on board telemetry generation process.

The actual software system, under development by Ciset in the frame of the Board Observability Breadboard project within the Hermes Programme (Ref. 4), will generate in real time telemetry packets filled with measurement values varying according to pre-defined variation laws. The telemetry generator is interactively commanded by test operator directives issued according to a telemetry plan, and sends the generated packets to communications simulator for space to ground link modelling. The simulator implements on board recording and packets playback functions, together with the filling of high rate telemetry with dummy packets when required.

The aim is to validate the software functional specifications with respect to the identified priorities for the S/W processes and to assess the overall system performances on the basis of the times needed for the execution of each elementary task.

TOWARD LOWERING THE COST OF MISSION OPERATIONS

S. D. Wall
 Jet Propulsion Laboratory
 Pasadena, CA 91109, U. S. A.

N94-23960

K. W. Ledbetter
 National Aeronautics and Space Administration
 Washington D. C. 20546, U. S. A.

ABSTRACT

The mission operations system is one of the more significant drivers of the cost of the mission operations and data analysis segment of missions. In large or long-lived projects, the MOS can also be a driver in total mission cost. Larger numbers of missions, together with an increasingly cost-conscious environment, dictate that future missions must more strictly control costs as they perform to their requirements. It is therefore prudent to examine the conduct of past missions for ways to conserve resources.

In this paper we review inputs made to past projects' "lessons-learned" activities, in which personnel from past projects (among other things) identified major cost drivers of MOSs and considered how economies were or might have been realized in both design and performance of their MOS. Common themes among four such reviews are summarized in an attempt to provide suggestions for cost reduction in future missions.

Key Words: Mission operations, cost efficiency, lessons learned

1. INTRODUCTION

Remote sensing missions, including both missions that explore beyond the terrestrial environment and those which use space as a vantage point from which to study earth, provide us a growing experience base in the design of mission operations systems (MOS).

The evolution of MOS designs has certainly been a benefit to the later projects, allowing them to concentrate more on research goals and somewhat less on the questions of how the basic spacecraft and mission will operate. This collective experience allows us to learn from our past successes and failures and gives us the possibility to reduce the operations costs of future missions from the lessons that the past provides.

In an attempt to document that experience base, several recent missions have conducted after-the-fact reviews of their mission operations, commonly known as "Lessons Learned" reviews. The purpose of such reviews has been to review the mission's performance and to record those elements that were either (1) done in a way worth recommending to future missions or (2) done in a less-than-optimum way and worthy of comment to allow future missions to correct the approach. Within the reports from these were many suggestions for improvement of cost efficiency in the mission operations system.

This paper results from an examination of Lessons Learned reviews for topics relating to cost. As such, it is a compilation of suggestions from the project team members who operate at the "worker" level as to how future projects might perform their tasks with higher productivity and lower overall cost. Inputs consisted of published Lessons Learned reports

from the UARS (Ref. 1) and Magellan (Ref. 2) projects, and less formal presentations and reports from Voyager and Galileo. Individual lessons that dealt with operations, ops management, or ops development were entered into a database and rated for their relevance to cost reduction. Suggestions were assigned to one or more of three mission phases: requirements definition; development, test and training; and operations. The database was then sorted for keywords, and common subjects were sorted according to the number of entries. This report is a compilation of the most often-mentioned subjects taken from that database.

Suggestions fell into two categories: those thought to increase mission success or to reduce risk, and those with potential contributions to reduction of cost. A total of 186 individual suggestions were related to cost savings. After sorting, 27 comments were found applicable to the requirements definition phase, 140 to development, test and training, and 90 to operations. In the following sections we summarize the content of the comments made concerning each phase.

2. REQUIREMENTS DEFINITION PHASE LESSONS

For the purposes of this paper we define the requirements definition phase as the period from project inception to the beginning of implementation, including any studies or preliminary definition steps. Chief among cost concerns in this phase were familiarity of both the details and the intent of requirements and continuing interaction with users to ensure that the requirements were well understood.

The most frequently mentioned topics relevant to the requirements definition phase suggested early achievement of a clear high-level

understanding of the goals, scope, and risk acceptance criteria envisioned by the project before beginning the process of writing requirements. This conceptual understanding should then be modified considering available resources. All projects in the study mentioned this issue. A similar set of comments suggested earlier involvement by representatives of spacecraft developers, science users, and operations designers in a mission definition document, which was variously referred to as a "Project Plan" or "Operations Concept" (Refs. 3, 4). The UARS document in particular recommended the development of a "realistic and affordable project plan" as a potential cost saver, specifying both local and remote facility designs as cost drivers to be understood early in the definition stage.

Next in number of recommendations in this phase was the subject of software requirements. UARS and Magellan suggested that software documentation such as Software Management and Development Plans and Users Guides be outlined prior to software definition to save redesign during implementation. Voyager suggested that earlier involvement of multimission representatives in requirements writing would save downstream costs. Magellan also cautioned that a cost-related tradeoff exists between the effort spent on completing requirements definition and that required for testing of partial software deliveries.

Recommendations from UARS and Magellan suggested that consideration of computer loading needs (e. g., disk space, CPU speed, and workstation sizing) as a fundamental requirement would have saved later expenditures to re-size processing systems that were specified without proper forethought. UARS also noted that ground data systems should be designed to accommodate expansion.

Magellan and UARS both commented that archiving requirements had not been sufficiently considered in the original levying of requirements, and that excessive resources were required to satisfy the projects' archiving obligations during the operations phase. Both projects benefited from a Data Management Plan which described archiving and dissemination designs, and UARS suggested that such a plan should be kept updated as the mission matures.

An unusual suggestion was made in the Magellan Lessons Learned that projects should as a matter of course perform an exercise following requirements definition wherein all requirements writers would be asked to identify those requirements that would be deleted should the project be required to halve its runout cost (an approximation of the descoping activity that created Magellan from its predecessor, VOIR). Such an exercise, done early in the design might be useful should actual descoping be necessary and, according to its author, could identify requirements that were less than critical.

Projects were unanimous in their belief that clear, concise requirements are a cost saver. Requirements must be written "clearly, completely and testably," and they must be "fully understood, identified and validated." More system engineering effort was recommended in the early phases of project definition. UARS and Magellan felt that system engineers needed authority over definition documentation, and that more science involvement would have avoided misstatement of resource requirements. These two projects also commented on the costs of technology development, and stated that proven system elements should be inherited, subject to several development caveats regarding inheritance of documentation and test data. Care was also advised to ensure that revision of

inherited software or hardware did not outweigh the original cost savings.

3. DEVELOPMENT PHASE LESSONS

We define the development phase as the time period beginning with production of a set of complete or nearly-complete requirements. In this phase the requirements are implemented by, for example, writing software, building flight hardware, or writing procedures and plans. Included in this phase are software and hardware test and integration and personnel training. The largest number of comments in the database concerned this phase.

Magellan, UARS and Voyager suggested that tests of various subsystems could be combined to save resources. UARS further suggested that coordination and more documentation of test requirements across support groups was indicated. Magellan and UARS both felt that earlier testing with dataflow, either real or simulated, would have saved overall costs. Training costs could be saved according to UARS and Magellan if simulators used for test and training could serve double duty as simulators for sequence validation during operations, and as substitutes for subsystems not yet ready in system-level tests.

Development-phase efficiency could also be improved with a clearer overall concept document, according to most projects. Specifically, Voyager referred to earlier involvement of multimission, project-specific operations and science personnel, and asked for more internal reviews. Magellan mentioned simultaneous design of mission, spacecraft and mission operations system, and UARS suggested that development of ground data systems should consider flight operations design.

Establishment and control of interfaces have been considered key to the effective MOS design (Ref. 4). Early identification of, characterization of, and agreement to interfaces was mentioned by three projects in this study. Identification of single points of contact for each interface was thought valuable by Voyager. Formal Software Interface Specification documents (SISs) were thought worthwhile by Magellan despite the unexpected effort required. Voyager also suggested that all parties be required to sign interface agreements. Magellan also indicated that many paper interfaces could be replaced by an electronic mail system.

The subject of change control was popular at Lessons Learned reviews. A tighter link between the entity that approves changes and the one that commits resources was mentioned by both Galileo and Magellan as a way of avoiding wasted time and thus funds. Galileo and UARS suggested that all requested changes to requirements be thoroughly evaluated by system engineers to avoid forgotten impacts. The Galileo project, which was forced to undergo several major design changes prior to its launch, indicated that changes must be documented and widely distributed when redesign occurs, and that the rationale for changes should be archived as well.

Another popular subject was that of the formal review process used by most NASA missions. Reviews were seen as necessary and beneficial by all projects, although the comment was made that the resources necessary for reviews need to be better anticipated. Independent design and analysis reviews were thought to be effective, as were design and interface "walkthroughs," and informal discussions prior to formal reviews. Involvement of flight controllers in sequence design reviews was thought by Voyager personnel to be an aid to avoidance of costly command errors.

Contingency plans are usually developed prior to beginning operations. The projects reviewed here produced varied opinions as to the tradeoff between development of a few contingency plans to a very high level (e. g., to the level of a command load that could be immediately uplinked) and development of a large number of contingency plans to a relatively low level. It was clear, however, that the level of plan development was considered to be a cost issue, and that the time spent in contingency work should be carefully balanced against acceptable level of risk.

4. OPERATIONS PHASE LESSONS

The operations phase includes all portions of the mission dedicated to actual acquisition of science data. Matters concerning science and science interfaces were most numerous among operations phase comments.

Both UARS and Magellan used a working-group approach for both data product and uplink interfaces with science teams and believed that approach saved time. Magellan believed that timely release of science data, both digital and photoproduct, to be effective, with a minimal required validation period. Magellan also suggested that science investigators' involvement in operations and data production was beneficial in reducing data production costs. Extensive use of students and post-graduates in operations was seen as a cost saver both at the operations center and investigators' home institutions.

Voyager, UARS and Magellan projects attempted some form of distributed operations to avoid the expense of transporting large numbers of investigators and/or contractor support personnel to the operations center. With few exceptions, the

response to such "remote" operations was positive. Voyager investigators were pleased with remote data access, but suggested that the access should be simpler and easier. Magellan urged future projects to emphasize better teleconferencing systems, more voice nets, and more regular status reports to ease the problems encountered with their remotely-located spacecraft team. UARS similarly commented that good communications and frequent meetings are beneficial. However, all three projects noted that physical separation of groups within a center was detrimental.

5. MANAGEMENT AND CONTRACTUAL LESSONS

Several comments in the Lessons Learned reviews pertained not to a particular phase of a mission but to contractual and management issues. Both Magellan and UARS felt that contracts written with award fees were more productive per dollar spent. Magellan added that to maintain productivity award fees should be constructed so that some incentive always remains for the contractor. Both projects also noted that time could be saved by working problems at the lowest possible management level.

6. CONCLUSIONS

Personnel involved in Lessons Learned activities were quite concerned about issues relating to productivity and cost-consciousness, and made many specific suggestions to future projects as to how resources could be saved or better directed. Surprising numbers of topics were found to be common among the four projects reviewed here.

According to our characterization scheme, the greatest number of comments received related to (1) software requirements placement and related development efficiencies, and (2) the achievement of a better conceptual understanding of the

mission before or early in the requirements definition phase. Issues regarding interfaces were second in number. Also receiving relatively large number of suggestions were distribution of facilities, importance of reviews and keeping a strong and responsible system engineering function.

7. ACKNOWLEDGEMENT

The authors would like to thank the Magellan, Voyager, Galileo, and UARS operations teams for their participation in the respective "Lessons Learned" activities, and the projects for providing us with the resulting reports. The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under contract with the National Aeronautics and Space Administration.

8. REFERENCES

1. UARS Project Team, 1992. The Upper Atmosphere Research Satellite Shared Experiences, Goddard Space Flight Center 743.1070892., Greenbelt MD
2. Ledbetter, K. W., and Wall, S. D., 1992. Magellan Lessons Learned: Proceedings of the Magellan Lessons-Learned Workshop, Jet Propulsion Laboratory Internal Document D-9643, Pasadena, CA.
3. Ledbetter, K. W. and Wall, S. D., 1992. An Operations Concept Methodology to Achieve Low-Cost Mission Operations, this issue.
4. Wall, S. D. and Ledbetter, K. W., 1991. Design of Mission Operations Systems for Scientific Remote Sensing, Taylor and Francis, London.

NEW MISSION REQUIREMENTS METHODOLOGIES FOR SERVICES PROVIDED BY THE OFFICE OF SPACE COMMUNICATIONS

Dwight P. Holmes and J. R. Hall

JPL, California Institute of Technology, Pasadena, CA 91109;

William Macoughtry, NASA/Goddard Space Flight Center,
Greenbelt, MD 20771; and

Robert Spearing, Telos Corp. Chantilly, VA 22021

N 94 - 23961

ABSTRACT

The Office of Space Communications, NASA Headquarters, has recently revised its methodology for receiving, accepting and responding to customer requests for use of that office's tracking and communications capabilities. This revision is the result of a process which has become over-burdened by the size of the currently active and proposed missions set, requirements reviews that focus on single missions rather than on mission sets, and negotiations most often not completed early enough to effect needed additions to capacity or capability prior to launch.

The requirements-coverage methodology described is more responsive to project /program needs and provides integrated input into the NASA budget process early enough to effect change, and describes the mechanisms and tools in place to insure a value-added process which will benefit both NASA and its customers. Key features of the requirements methodology include the establishment of a mechanism for early identification of and systems trades with new customers, and delegates the review and approval of requirements documents to NASA centers in lieu of Headquarters, thus empowering the system design teams to establish and negotiate the detailed requirements with the user. A Mission Requirements Request (MRR) is introduced to facilitate early customer interaction. The expected result is that the time to achieve

an approved set of implementation requirements which meet the customer's needs can be greatly reduced. Finally, by increasing the discipline in requirements management, through the use of baselining procedures, a tighter coupling between customer requirements and the budget is provided. A twice-yearly projection of customer requirements accommodation, designated as the Capacity Projection Plan (CPP), provides customer feedback allowing the entire mission set to be serviced.

1. INTRODUCTION

The Office of Space Communications is the programmatic office responsible for providing spacecraft operations and control centers, ground and space communication, data acquisition and processing, flight dynamics and orbit determination, and spacecraft tracking services for NASA's customers. The Office is currently going through an evolution in the way business is conducted and services provided to its customers. The Office is responsible for managing all of the tracking and data acquisition resources of NASA. As such, the Office of Space Communications manages: 1) the Space Network which consists of three fully operational and two partially operational, Tracking and Data Relay Satellites (TDRS) in geosynchronous orbit and the vestigial elements of the STDN Ground Network, both operated by the

Goddard Space Flight Center; 2) the Deep Space Network operated by the Jet Propulsion Laboratory and; 3) facilities for providing communications services for aeronautics, balloons, and sounding rocket programs operated by the Wallops Flight Facility (WFF), Dryden Flight Research Facility (DFRF), and the Ames Research (ARC) Center, (Moffett Field and Crows Landing) .

In providing services to its customers, the Office of Space Communications is streamlining the evaluation of requirements and delegating the generation of the required detail as well as the responsibility of responding to those requirements to designated lead NASA centers.

2. COMMUNICATIONS SERVICES

Communications services are technically provided through designated lead centers. The Jet Propulsion Laboratory (Deep Space Network) provides tracking, communications (telemetry and command) and navigation services for Deep Space missions, highly elliptical Earth orbiters, missions at Lunar distances, and for non TDRS compatible low Earth orbiters. The Goddard Space Flight Center (Space Network, Ground Spaceflight Tracking and Data Network, and Wallops Flight Facility) for TDRS compatible Earth Orbiters, Low Earth orbit and suborbital tracking, communications, navigation, and certain control center and data processing functions. Wallops Flight Facility also provides tracking, communications, navigation and command services for low Earth orbiters, highly elliptical orbiters, sounding rockets, balloons, and portions of the aeronautics programs. Dryden Flight Research Facility provides services primarily for the aeronautical missions and Space Transportation System (STS - Space Shuttle) return activities. Moffet Field and Crows Landing (ARC) also provide services

for aeronautical missions.

3. NASA'S CUSTOMERS

Customers include NASA flight and non flight missions as well as other civil government space operations, the Department of Defense, Commercial and Foreign missions (government and commercial). Top priority for the allocation of Tracking and Data Acquisition resources is allocated to NASA missions. In addition, NASA cooperative experiments and services and non-NASA missions are supported. This coverage approach is by design, in that NASA, as its own customer, defines the resources for the tracking and data networks. Other users who conform to the standardized capability provided can also be serviced. There is usually no implementation and/or modification that is driven by non-NASA customers. In the current environment this poses a challenge to the requirements process. There are increasing numbers of non-NASA users. Multinational programs in which NASA participates as an equal partner, and uniquely foreign programs, which require NASA communications resources in order to provide a viable mission, must be covered. Requirements for these new missions continue to push the limits of both the capability and capacity of the ground and space networks. Further, there has been a greater emphasis placed on commercial space activities which most likely depend on the NASA tracking and data acquisition network infrastructure. NASA, of all the worlds space agencies, maintains the most capable set of networks with the greatest capacity. For example, to date, there is no other Space Network providing complete near Earth tracking and data relay services. However the National Space Development Agency (NASDA) of Japan and the European Space Agency (ESA) are expected to develop this capability in the near future.

4. REQUIREMENTS ENVIRONMENT

Prior to the spring of this year, the requirements management process at the Office of Space Communications had been directed toward "short term" solutions. In particular, there was no systematic procedure for early notification of a customer's needs. The organization responsible for providing the services would not be informed of the real requirements until later mission phases and usually, therefore, too late to provide the essential services needed to meet those requirements. Also, the roles and responsibilities for both Headquarters and the lead centers were not clearly defined. Approvals on requirements documentation would occur after the fact, and baselines for requirements with their associated budgets were not always clearly established.

New methodologies for requirements management will address these issues and demonstrate plans for improving responsiveness to meet the customers needs for both communications service capacity and capability.

5. THE FIRST STEP

Recognizing the inadequacies of the current process, the Office of Space Communications, with recommendations from the Office of Space Science and Applications (OSSA), decided to revise the NASA Management Instruction (NMI 8430.1B), "Obtaining Use of Office of Space Communications (OSC) Capabilities for Space, Suborbital and Aeronautical Missions". The resulting document, (NMI 8430.1C), signed by the NASA Administrator last December 31, 1991, addresses the issues of the previous system and is the vehicle for establishing a new process. The revised NMI promotes and fosters early identification of customer requirements in part by conducting periodic

planning activities with customers. The new process requires notification of requirements at the completion of the Phase A studies. In most cases this provides a four to five year lead time before services are actually needed. The document for such notification is called the Mission Requirements Request (MRR), which unlike earlier documentation is a maximum of seven pages. The format facilitates the capture of the contents into a requirements data base lending itself to the identification of cost and performance "tall poles". The opportunity for establishing a centralized requirements tracking system and a comprehensive mission requirements data base is an important aspect of the new process.

Another key provision of the new instruction includes periodic feedback to the customer indicating the degree to which his requirements can be accommodated. This is accomplished by way of a Capacity Projection Plan (CPP). The CPP provides the expected capacity and capability needs and costs to support the customer mission set for the next five years. The periodicity of the CPP is, at a minimum, coincident with the budget cycle. Having a periodic plan which looks at resources and the capacity to provide services to customers representing the entire mission set allows planners to make appropriate reallocations within the budget. The allocation can be for either new capability and/or increased capacity which can best fit the customers needs or for reducing mission requirements. This process requires mission-coverage trade offs. Customers may not always be able to acquire all the coverage their original mission needs. Thus the CPP provides the negotiating baseline for working potential shortfalls for the next budget cycle. The CPP provides a mechanism for tightly coupling requirements management to the budget process. The CPP requires strong interaction from all participating OSSA and OSC lead centers and forces discipline in the

requirements management process. The detailed CPP process infrastructure is currently being developed.

Finally, the revised NMI delegates much of the responsibility for negotiating the detailed requirements between cognizant mission and OSC lead centers. The Detailed Mission Requirements (DMR) document is a negotiated document between the service provider and the customer and provides the baseline for performance, schedule, and cost. It contains both the requests and the responses to those requests. Previously that was accomplished with two documents, the Support Instrument Requirements Document (SIRD) and the NASA Support Plan (NSP). Authority to provide the services requested and negotiate the detailed requirements with the customer rests on the success of early design and cost planning at the lead center. This is particularly true for those requirements that can be satisfied within projected capability and capacity. The Office of Space Communications will be kept informed of the activity but is not heavily coupled to the authorization loop. By placing the responsibility for satisfying the detailed requirements down where the services are provided, the response time is shortened, allowing much greater freedom to negotiate how those requirements are satisfied.

When it is determined at the lead center that the customer's requirements cannot be satisfied within the projected capability or capacity and network augmentation is required, the budget impacts are forwarded to the OSC for appropriate action.

6. THE DETAILED REQUIREMENTS MANAGEMENT PROCESS

As mentioned earlier, the requirements management process is evolutionary. This paper discusses where that process has gone during the past number of months. The

process is still in the formative stage. The development of a Capacity Projection Plan process is still under way and requires the joint negotiation of all the lead centers. The Mission Requirements Request form currently in use remains in draft format. As the CPP is developed there will be room to fine tune the MRR to best fit the data requirements of the CPP. The formal response to the projects via the CPP on how mission requirements are accommodated will not be available for at least another six months. However, during the development of the first capacity projection plan, that response is provided by ensuring close communication between the users and the providers. This process can succeed if adherence to the principles which were established at the onset of this process are followed. Those principles are:

1. Early notification of coverage and performance requirements.
2. Development and costing of capacity and capability options.
3. Delegation of the responsibility for authorizing the service at the lead center responsible for providing the service.
4. Evaluation of all the resources for the accommodation of requests for services based on the entire mission set.
5. Tight coupling of the budget process to the request for services, including modifications of those requests.

The first step in satisfying a customers requirements is early knowledge of those requirements. Ideally, this is accomplished during the early Phase A mission study process. Early notification allows mission and OSC system design teams to develop the architectural option and results in the Office of Space Communications developing

the necessary long range plans, network loading projections, facility design, performance, and cost data to the new mission. The user, along with notification of intent to use OSC resources, also identifies any unique capability that may be required. All of this sets the stage for the development of the Mission Requirements Request. The MRR is generated as a result of flight-ground teaming relationships from the very beginning of a project. In fact, during the Phase A study period, the candidate OSC lead center, whose responsibility it will be to provide the service, provides loading impacts, its long range plans, and options. At the start of Phase B, the generation of the MRR is completed with all the information understood at that time. The MRR formalizes the "tall pole" requirements but does not represent the final negotiated requirements. The intent is to force early notification and identification of the important aspects of the requirements. The MRR evaluation continues throughout the design phases of the project.

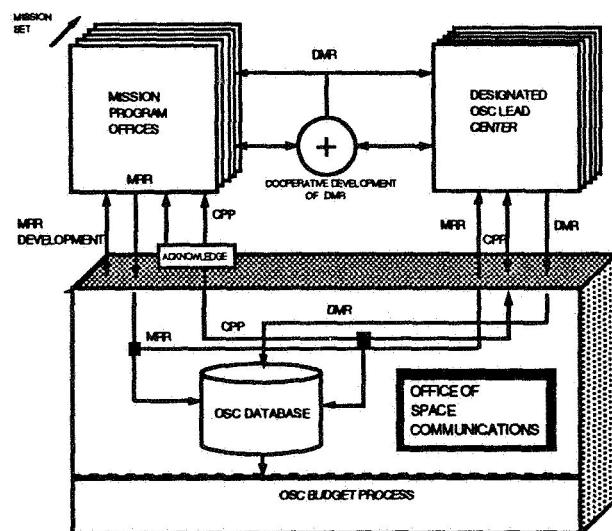


Figure 1. Process Relationships

The formalization of the MRR, as signified by a signed letter of transmittal by the mission program office, allows the

responsible organizations to identify manpower and other resources required to work the requirements. The conclusion of a Phase A study generally results in the authorization to proceed as a new start. If approval is not obtained, the MRR is put on hold. If a MRR has been forwarded, it is withdrawn by the project to be reinitiated at a later date if the project becomes approved.

Figure 1. graphically represents the relationship of activities in the process. Many of the processes are interactive and are signified by arrows in both directions. The relationship between the OSC and the organizations representing each of the missions is dominated by the MRR process. The formal MRR is symbolized by a unidirectional arrow from the program office. The formal MRR is acknowledged by OSC and the lead center is identified. The identification of the lead center authorizes the start of the development of the DMR and any processes associated with the development of the CPP. The right side of the graphic represents those relationships with the lead centers. However, the relationships are not limited to those depicted. For example, the OSC provides supporting information on the technical capabilities of the networks as well as coverage plans.

Acknowledgement signifies another milestone in the requirements process. A mission set data base is updated with MRR, DMR and CPP data. Impacts to other missions, OSC resources in terms of capacity and capability, and the budget are extracted from the data base. As depicted, the data base is a compilation of knowledge from the mission and OSC lead centers. The current development on the database uses the MRR as fundamental requirements element with supporting data and links to more detailed documentation.

The Detailed Mission Requirements document reflects the MRR and, as the name

implies, identifies the detailed requirements of the mission. But more than that, the DMR is a cooperative document between the OSC lead center and the customer's program office. Figure 1. depicts the level at which the DMR is developed. The DMR is a joint document between the mission and OSC lead centers, but it also serves a feedback mechanism for OSC as to the status of requirements assignment or implementation. Where as, earlier documentation was structured as "request then response", the DMR integrates the mission requirements into lead center's plan for satisfying those requirements into a single document. The result is more akin to a negotiated agreement on the services to be provided. The DMR is produced during the Phase B cycle in the mission design process and the requirements are traceable to the MRR

Even though the plan is to create a formal CPP which is published twice yearly, the CPP itself is continually updated based on mission requirements as they evolve. Impacts and changes to the CPP are evaluated and negotiated with impacted projects and the CPP assessed against budgetary plans and constraints. The key to the process at this point is consideration of both mission sets and priorities. Earlier processes tended to make judgements of supportability on a mission by mission basis. The current process provides for evaluation of services to NASA's customers on the entire collection of currently supported missions and authorized plans. This concept allows evaluation and cost tradeoffs which can provide services to the greatest number of missions and reduce the costs to NASA. Requirements tradeoffs are made early in the mission design and are used to partition the requirements such that the design of the communications link, or other resource, is an equitable share mission design team and the lead support network. Mission sets also provide the forecaster the luxury of defining trends in

capacity and capability to support classes of missions, thereby simplifying the process of generating a projection plan.

The MRR, DMR and CPP are living documents subject to change. This process has taken change into account and made that part of the system. By recognizing that mission events and plans are dynamic, this process will allow the responses to changes in plans to take place effectively and in a timely manner. The figure depicts a continuous process, and not one with milestones fixed to arbitrary events.

7. SUMMARY

The Office of Space Communications, NASA Headquarters, has recently revised its methodology for receiving, accepting and responding to customer requests for use of that office's capabilities. The methodology described in the paper is both more responsive to project/program needs and provides integrated input into the NASA budget process early enough to affect capacity and capability change. Important aspects of the methodology include its focus on network capacity integrated across the entire mission set and its ability to identify early, new requirements for added capacity and/or capability.

8. REFERENCES

NASA Management Instruction, NMI 8430.1C "Obtaining Use of Space Communications (OSC) Capabilities for Space, Suborbital and Aeronautical Missions", December 31, 1991

Acknowledgement; The authors wish to acknowledge the help provided to us by members of the Requirements Branch, Program Integration Division, Office of Space Communications, and to lead center personnel at Goddard Space Flight Center and the Jet Propulsion Laboratory.

FROM REQUIREMENTS TO ACCEPTANCE TESTS

Lionel BAIZE - Hélène PASQUIER

N 9 4 - 2 3 9 6 2

C.N.E.S. - TE/IS/PS/SI
18 av. E. Belin
31055 TOULOUSE Cedex - FRANCE

Abstract

From user requirements definition to accepted software system, the software project management wants to be sure that the system will meet the requirements. For the development of a telecommunication satellites Control Centre, C.N.E.S. has used new rules to make the use of tracing matrix easier. From Requirements to Acceptance Tests, each item of a document must have an identifier. A unique matrix traces the system and allows the tracking of the consequences of a change in the requirements.

A tool has been developed, to import documents into a relational data base. Each record of the data base corresponds to an item of a document, the access key is the item identifier. Tracing matrix is also processed, providing automatically links between the different documents. It enables the reading on the same screen of traced items. For example one can read simultaneously the User Requirements items, the corresponding Software Requirements items and the Acceptance Tests.

Keywords : Requirements, development, tracing matrix, tool.

1. DOCUMENTATION MANAGEMENT

1.1 Identification of requirements

The problem of identifying precisely the requirements when developing software, is

known by any one who has had to manage a subcontracted development on behalf of a final client : the operational team.

One expects the project manager to answer such question as "what about this requirement, are you sure the modification we wanted has been correctly taken into account,... ?".

Thus to avoid ambiguity and inaccuracy, for the development of a telecommunication satellites control centre, C.N.E.S. decided as early as 1989 to introduce identifiers in the User Requirement Documents.

1.2 Identifiers

In order to have a representative identification (trying to avoid FRED1253), it was decided that an identifier must contain the acronyms of the main functionality concerned (TM, TC,...) and of the requirement, followed by a number, e.g. TMVISU090 for a requirement dealing with telemetry visualization.

1.3 Editing rules

In order to have a tool based on the use of a **DataBase Management System**, we chose editing rules to allow an automatic feeding of the database. To be followed the rules have to be simple :

- an identifier must be preceded by a carriage return character and followed by a tabulation character,
- when a modification is made to the requirement, the identifier must be preceded by a "£" character in the next version of the document.

2. THE DOCUMENT DATABASE

2.1 Definition of the database

The need at this stage was to gain an automatic access to an identifier, the main key of the database.

But we also wanted to know :

- which document it comes from,
- which edition/revision of the document,
- which paragraph (number and title) containing it,
- the text of the requirement.

For a requirement subject to a change we also wanted the change request number which it originated from.

2.2 The database

With 4^e DIMENSION® which is a user friendly DBMS running on a Macintosh® we have developed a tool named "TBD-pro" for "Traceur de Besoins durant le Développement - version probatoire".

TBD-pro was designed and developed at C.N.E.S. by M. Studnia, P. Pacholczyk, L. Baize.

To avoid to be dependent on the word processor used to edit the document, the automatic feeding of the database is done on ASCII files. For the first "import" of a document, you create as many records as identifiers, for the next ones you only import into the database the requirements whose identifiers are flagged with "Σ", the previous records of the database with the same identifier are flagged as invalid.

In order to be able to trace the user requirements with other documents, we followed the same syntactic rules for the Software Requirement Document and for the System Acceptance Tests Document which were added to the database.

FONCTIONS	
Référence	SPACEOPS010 Etat ☐ Valid ☐
Synthèse	This is an abstract of the requirement
Autre Référence	SPACEOPS020 DM ☐ 1
Nomenclature	SP-URD-001 Edition ☐ 1 Révision ☐ 0
Paragraphe	1.1. Introduction to T.B.D.-pro
Descriptif	Here is the very precise, testable second version of the user requirement which must illustrate the presentation of TBD-pro at the Second International Symposium on Ground Data Systems for Space Mission Operations.

3. THE TRACING DATABASE

3.1 Definition of the tracing matrix

We wanted to cross check the user requirements with the acceptance tests. The acceptance tests are a subset of the validation tests, which have to be checked against the software requirement. Thus it was decided to have a unique matrix containing the user requirements, the software requirements, the corresponding tests and the phases (validation/acceptance) the test is performed. Each identifier is followed by a tabulation character and a carriage return character separates each line of the matrix.

UR Ident.->SR Ident.->Test->PhasesCR

3.2 The database

One of the main problems one has, reading a tracing matrix, is that very often the identifiers are not explicit. Thus the only

way to study it, is to have on one side the matrix, on the other side the concerned documents.

As we had the document database, we decided to link the records with the matrix to offer a new matrix presenting the contents of the requirements or tests and not only the identifiers.

Thus a new version of TBD-pro was developed, to include this functionality.

T.B.D. pro	1 2 3 4 Fiche N°: 017 Imprimer Quitter	T.B.D. pro
REFERENCE SPACEOPS010 MINÉMONIQUE FRD		
PARAGRAPHE 1.1. Introduction to TBD-pro		
SYNTHÈSE This is an abstract of the requirement		
DESCRIPTION Here is the very precise, testable second version of the user requirement which must illustrate the presentation of TBD pro at the Second International Symposium on Ground Data Systems for Space Mission Operations		
REFERENCE SOFTOPS030 MINÉMONIQUE FRD		
PARAGRAPHE 2.1 Presentation of the software system		
DESCRIPTION Here is the very precise, testable second version of the software requirement which must illustrate the presentation of TBD pro at the Second International Symposium on Ground Data Systems for Space Mission Operations		
REFERENCE TESTOPS020 MINÉMONIQUE FRD		
PARAGRAPHE 1.1 Test of the MMI		
DESCRIPTION Here is the validation Test which must illustrate the presentation of TBD pro at the Second International Symposium on Ground Data Systems for Space Mission Operations, testing SOFTOPS030.		
PHASES 5.4		
COMMENTAIRE This test procedure was developed during the 1995/96 validation phase		
FICHE N° 051		

4. UTILIZATION OF TBD-pro

4.1 Completeness of the matrix

Because the documentation is in the database, TBD-pro offers the automatic check of the completeness of the matrix. You may print and/or store in a file the missing requirements.

4.2 Searches in documents

TBD-pro offers the capability of searching through the documents. For instance, one may :

- search requirements involved in a precise change request,
 - rebuild an out of date version of a document,
 - combine searches e.g. one may look for requirements containing strings like "telecommand" and "operator" and not "alarm",
- and then store and/or print the result.

4.3 Other facilities

TBD-pro offers the capabilities of searching through the documents on top of searching through the tracing matrix.

One may perform any search he wants through the documents, and get onto the screen the corresponding selection of requirements and/or tests from other documents.

4.4 Performances

Our User Requirement Document contains 1100 requirements, whole import takes less than one hour.

The full matrix contains about 4000 lines. It takes about two seconds to change from one line of the matrix to the next one.

Checking that no identifier from a document has been omitted takes about one hour and a half.

4.5 Planned upgrades

Up to now, the version of TBD-pro is an interpreted one. In order to gain rapidity, it will be updated, to be compatible with the latest version of 4^e DIMENSION®, and compiled.

To be widely distributed, it needs to be an industrial product with a real

documentation and not only some developers' notes.

5. CONCLUSION

TBD-pro has shown it could be very helpful, because it is exhaustive, simple to use (menus, the result of any search may be stored and/or printed). It is the technical configuration management tool of the project.

Macintosh is a registered trademark of Apple Computer, Inc.

4^e DIMENSION is a registered trademark of ACI.

RESOURCE ALLOCATION PLANNING WITH INTERNATIONAL COMPONENTS

Gene Burke
JPL, Multimission Operations Systems Office
Ralph Durham
JPL, Resource Allocation Planning
Frank Leppla
JPL, Resource Allocation Planning
David Porter
California Institute of Technology, Economics Faculty

N94-23965

ABSTRACT

Dumas, Briggs, Reid and Smith (1989) describe the need for identifying mutually acceptable methodologies for developing standard agreements for the exchange of tracking time or facility use among international components. One possible starting point is the current process used at the Jet Propulsion Laboratory (JPL) in planning the use of tracking resources.

While there is a significant promise of better resource utilization by international cooperative agreements, there is a serious challenge to provide convenient user participation given the separate project and network locations. Coordination among users and facility providers will require a more decentralized communication process and a wider variety of automated planning tools to help users find potential exchanges. This paper provides a framework in which international cooperation in the utilization of ground based space communication systems can be facilitated.

Key Words: Resource Allocation Planning, Ground Data Systems, International Collaborations, Scheduling

1. INTRODUCTION

Dumas, Briggs, Reid, and Smith (1989), hereafter identified as DBRS, forcefully argued for increased efforts to facilitate, through more routine agreements, international collaboration in Deep Space communication and tracking. They pointed out that the scope of international cross-support has expanded over the years due to the increased trend toward international missions (eg. Ulysses and TOPEX) and international investigations and instruments on spacecraft (eg. Cassini and Huygens probe). The benefits, identified by

DBRS, of international coordination in which ground networks are used by other agencies and nations (this is defined as *cross-support*) include:

- The expanded use of arraying to increase receiving capabilities
- The progression toward very long baseline interferometry (VLBI), and
- The science for tracking exchanges that have allowed increased data capture for high activity periods and continuous coverage by agencies that lack world-wide coverage.

In order to facilitate the planning of international resource exchanges and cooperation, DBRS suggest that rather than the current mission-by-mission ad hoc development of cross-support agreements, exchanges of tracking time or facility use should become more standardized. In addition, active participation by all users and networks through better communication and software tools would provide a quick and convenient method to examine the range of trading options, and therefore expanding the amount of data capture from deep space missions.

The Consultative Committee for Space Data Systems (CCSDS) is an international interagency organization that provides a forum for space agencies to develop recommendations on standard techniques for data handling. The CCSDS "green book" provides a catalog of all available tracking facilities and their basic characteristics. Through this committee, significant headway has been made by the operations portion of the cross-support venture in identifying available resource capabilities. However, while much work has been done on

improving data standards and communication frequencies among ground data systems, there has not been much effort in standardizing cross-support agreements and integrating plans. The planning portion of the cross-support equation has not been well developed. Currently, there does not exist a common database in which individual agency network plans can be viewed. Thus, the ability to identify exchanges that can increase the amount of tracking for all agencies is not an easy task. Indeed, routine attempts to find beneficial trades of tracking time are not possible given the current structure of planning databases and software tools.

The purpose of this paper is to expand on the theme developed by DBRS focusing primarily on integrating and "standardizing" planning databases, so that beneficial trades can be found among participating networks. The same effort that has been undertaken to standardized data handling should be extended to the requirements and planning databases. This effort would then allow for simple standardized tracking exchanges to occur. The recommendations contained in this paper focus on the following four items:

1. A standardized requirements database for all participating "agencies" should be accessible by all planning teams and users.
2. Viewperiod data and mission set information should be made accessible to all users and planning teams.
3. Current resource plans and activities/events should be standardized and easily accessible.
4. Possible process flow of international exchanges.

If the planning databases of each agency can be "standardized", then the planning analysts for each agency will be able to more readily identify likely candidates for feasible cross-support exchanges. Once databases are standardized, then the next step would be to construct simple arrangements that can implement exchanges.

2. RESOURCE ALLOCATION PLANNING (RAP) PRIMER FOR DEEP SPACE NETWORK RESOURCES

In this section we will provide a brief overview of how resource allocation planning is conducted for NASA at JPL. The purpose of this primer is two-fold:

1. Describe the basic data structure of the JPL planning database; and
2. To provide a starting place for developing planning interfaces for international cross-support activities.

The process of matching spacecraft tracking requirements with available NASA tracking facilities has evolved from a pencil and paper activity to its present state as an semi-automated computerized system with look-ahead planning algorithms (see Berner et al (1989), Johnson and Werntz (1987)). The current system allows for plans extending over ten years (see JPL-RAP summary document). The RAP process provides allocation plans for tracking resources (antenna time) from 8 weeks to 10 years prior to execution. The plans range from generic levels of mission tracking support to detailed minute by minute track assignments. The RAP process has the ability to identify high activity-contention periods early (5-7 years in advance) and maintains a centralized database of requirements, major events, allocations and spacecraft viewperiods. The RAP process relies on four major components to forecast and plan resource use for NASA's Deep Space Network (DSN):

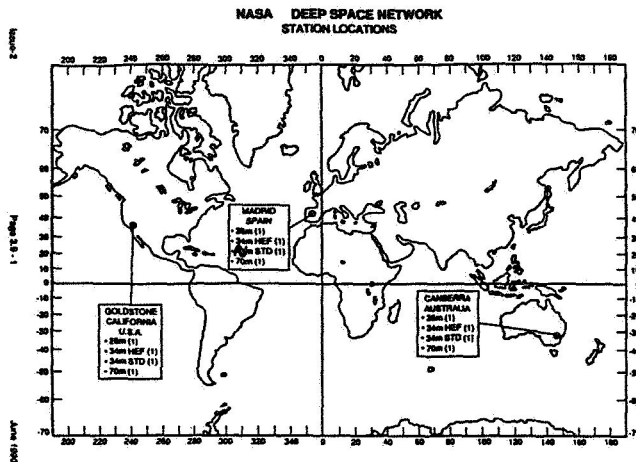
1. Network resource availability;
2. User requirements/requests and major events;
3. Planning constraints; and
4. Planning control.

2.1 Tracking Resources

JPL's Office of Telecommunications and Data Acquisition (TDA) maintains the configuration and operating policy for the DSN resources pictured in Figure 1. Information concerning the capabilities of the network, future plans, modifications and maintenance activities are provided to the Resource Analysis Team (RAT) by TDA.

Given the network plans and maintenance requirements, a profile of antenna location, availability and capabilities can be constructed. The resource profile consists of the number of available tracking hours at each station. Within the resource profile mission requirements are then planned.

Figure 1
DSN Resource Configuration Map



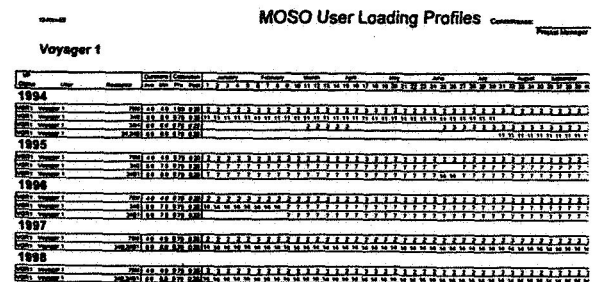
Source: CCSDS Green Book2.2

2.2 User Requirements

In addition to DSN maintenance requirements, each mission submits a 10 year plan of requirements and major events to the planning group. The set of missions used to develop a resource allocation plan is provided by a mission set that is developed by the RAT from NASA headquarters inputs. The current list of users in the JPL Resource Allocation Planning database can be found in Appendix A. The initial set of requirements submitted by users are generic in nature and list the number of "passes" per week (including pre and post track calibration requirements) and antenna requirements (eg. X-band, 70M) to support their mission. A pass is typically an 8 hour track but it can be shorter or longer depending on the tracking event. In addition, special requests such as overlap coverage from two stations can be requested along with spacing between passes can be requested. Figure 2

supplies a requirements input from a JPL mission. The input form lists, for each year, subnetwork, (we define a *subnetwork* as the system of either the 34 meter standard antennae (34S); 34 meter high efficiency antennae (34H); the 34 meter beam wave guide antennae (34B); or the 70 meter (70M) antennae) and week (1-52), the number of passes requested. For example, in Figure 2, in 1998 this mission is requesting 14 passes a week on either the 34S or 34B subnetworks.

Figure 2
User Requirement Form



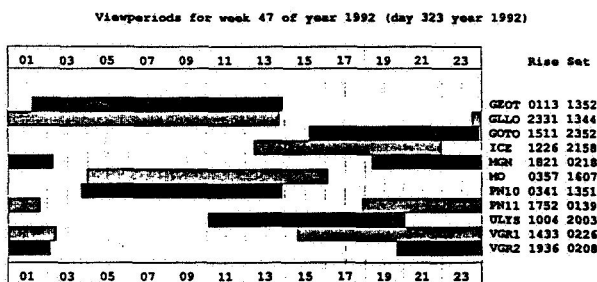
Source: JPL-RAP Database

The user requirements input become part of the requirements database that is used to produce an initial plan of network use. Specifically, given the inputs from users, a look ahead algorithm places user tracks into the plan so that average contention is "minimized" (see Johnson and Werntz (1987)). Given the average profile of use, high contention periods can be identified and effort can be directed at developing solutions to reduce the contention during the high activity periods. From the generic inputs and average contention profile, a detailed plan for use can be generated from which specific allocation conflicts can be identified.

2.3 Constraints and Events

One of the major drivers in the planning of DSN use is the time in which a spacecraft can be tracked by a station. This location-time element is defined as the spacecraft *viewperiod*. In the planning of network use, the viewperiod constrains the location in the schedule where a mission pass can be placed. Figure 3 supplies a portion of the viewperiod database for the Goldstone complex for missions in the JPL-RAP. The viewperiod data shows the exact mission times that are in view of the complex for a specific week, day and year.

Figure 3
Mission Viewperiods at Goldstone Complex



Source: JPL-RAP database

In addition to the viewperiod constraint, major events (eg. launch, orbit insertion) are also supplied with user requests so that the planners and users can view, in the plan, the events that are associated with the requests schedule in the plan. The combination of user requirements, mission events, and viewperiod constraints are integrated to provide a 10 year plan. Figure 4 provides a snapshot for the 70 meter network for part of 1994. In Figure 4 each user of the 70 meter network is listed along with its requirements (passes per week) and major events.

Figure 4
70M Snapshot of Resource Use Plan-1994

USER LOADING PROFILE - 1994

Major Events						MARS OBSERVER SUPERIOR CONJUNCTION														
						* ISTP WIND LAUNCH * 3/9/94 VGR 2 SC														
						1/30/94 2/13/94														
						DSS-14 CABLE WRAP REPAIR														
						3/21/94 4/3/94														
						DSS-63 CABLE WRA														
VP	User	Resource	Durations		Calculation	January			February			March								
Object			Ave	Min	Pre	Post	1	2	3	4	5	6	7	8	9	10	11	12	13	14
GLLO Galileo		70M	8.0	8.0	1.50	0.25														
STRN GSSR		DSS-14	8.0	8.0	1.50	0.25														
MERC GSSR		DSS-14	8.0	8.0	1.50	0.25														
MARS GSSR		DSS-14	8.0	8.0	1.50	0.25														
JUPIT GSSR		DSS-14	8.0	8.0	1.50	0.25														
NONE GSSR Antenna		DSS-14	8.0	8.0	1.50	0.25														
ICE ICE		70M	4.0	4.0	0.75	0.25														
ICE ICE		70M	8.0	8.0	0.75	0.25														
DSS Maintenance		70M	8.0	8.0			6	6	6	4	4	6	8	6	6	6	4	4	6	6
MO Mars Observer		70M	8.0	8.0	1.25	0.25														
MO Mars Observer DOR		70M	4.0	4.0	1.00	1.00														
PH10 Pioneer 10		70M	8.0	8.0	0.75	0.25														
PH10 Pioneer 10		70M	10.0	10.0	0.75	0.25	14	14	14	14				14	14	14	14	14	14	14
PH11 Pioneer 11		70M	8.0	7.0	0.75	0.25														
PH11 Pioneer 11		70M	7.5	7.0	0.75	0.25														
NONE Radio Astronomy		70M	8.0	8.0	0.75	0.25														
NONE VLBI Clock Sync		70M	3.0	3.0	1.50	0.50														
VGR1 Voyager 1		70M	4.0	4.0	1.00	0.25														
VGR2 Voyager 2		70M	4.0	4.0	1.00	0.25														

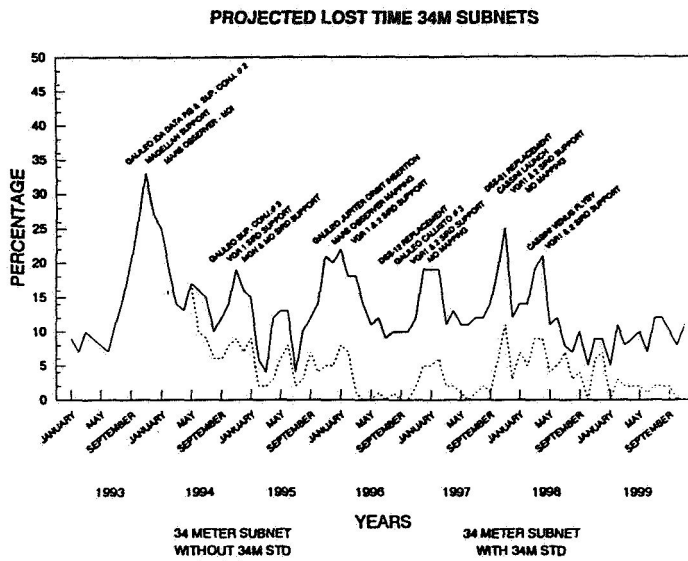
Source: JPL-RAP database

2.4 Planning Committees

Control of the resource plan and user requirements are provided by a three-tier committee process. At the top tier there is a Resource Allocation Review Board (RARB) which oversees the ten-year plan and user requirements. The Board consists of the project managers and scientists for each mission in the mission set, and meets biannually. The Board resolves the high-level contention and controls user requests and requirement changes, i.e., it provides for configuration control. For example, Figure 5 provides an overview of the percent of total 34 meter requirements (in hours) that cannot be met given the viewperiods (lost), users requests and system capabilities. The solid line shows the "lost time" from 1993 to 1999 for the entire 34 meter subnetwork if the 34S subnetwork is decommissioned when the 34B subnetwork is installed. The graph lists the major events at the spiked points. The graph also shows the benefits in terms of reduced lost time if the network is not decommissioned (see the dashed line). These types of graphs are becoming

standard output from the the JPL-RAP database. They provide a quick summary of the contention points in the schedule to be resolved. In addition, the database can allow for "what if" analysis by adding and subtracting resources from the system.

Figure 5
Analysis of Lost Time



Source: Resource Allocation Review (August 4, 1992)

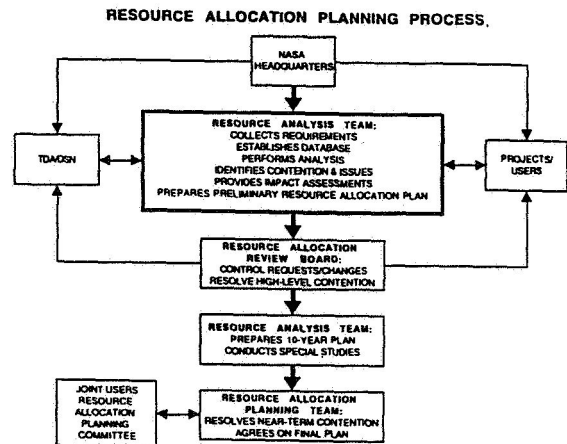
The second tier committee is the Joint Users Resource Allocation Planning (JURAP) committee which is comprised of flight project operations managers, DSN operations managers and other user operations managers. The JURAP is responsible for implementing the RARB recommendations and maintains requirements control as request become more specific and conflicts become more detailed. The third tier committee is the Resource Allocation Planning Team (RAPT) which consists of representatives from each mission user as well as DSN operations. The RAPT resolves contentions concerning specific track-by-track conflicts that are not picked-up in the other committees in their deliberations.

2.5 Process Flow Summary

Figure 6 provides an overview of the flow of the RAP process. The major elements of the process are:

- The mission set defining the current and potential users of tracking resources.
- A detailed list of network plans, characteristics and maintenance requirements which provide a resources availability profile.
- A standardized requirements input form for "generic" passes per week for each user.
- A look-ahead algorithm that places passes in the schedule so as to minimize average contention.
- A standardized set of outputs that allows planners to perform impact studies on various changes in the plans.
- A multi-tier committee process that provides for request control as the plans mature.

Figure 6
RAP Process



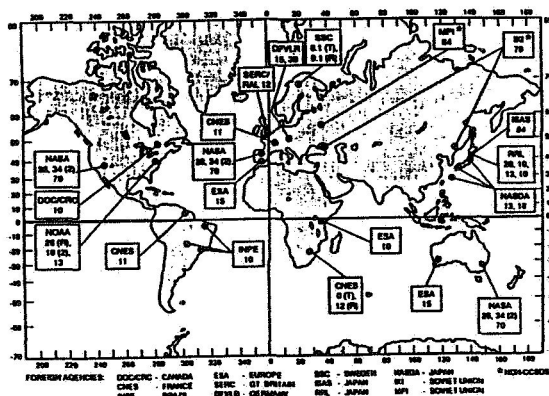
3. INTEGRATING INTERNATIONAL COMPONENTS

The purpose of this section is to show how elements of individuals plans can be integrated into a cross-support exchange system. It is axiomatic that more requirements can be met if tracking resources can be mixed from a variety of locations and agencies. Identifying potential mixes is where the use of common and standard databases can most helpful.

3.1 International Tracking Resources

The CCSDS maintains a catalog of all international tracking stations and their characteristics. This information is contained in the green book and from which Figure 7 below is taken from. This catalog provides a complete reference for each of the operating entities for each agency. For purposes of resource planning, additional information on each network's capacity and maintenance requirements, along with a profile of resource availability should become part of the JPL-TDA database that can be accessed by the JPL-RAP database. This would provide an ability to determine resource availability for different networks in each users viewperiod.

Figure 7
International Antenna Locations



Source: CCSDS Green Book

3.2 Requirements

Each network's user requirements should, to the maximum extent possible, be standardized so that plans and forecasts can be interpreted easily. Specifically, for the longer range plans from 3-10 years, generic "passes" per week for users could be the norm unless some other message form naturally emerges. Such standards could be developed through an international joint committee on network resource planning. This committee would likely consist of planning analysts whose agencies agree like to participate in standard cross-support arrangements.

With a standardized requirements input format each agency can easily examine the requirements database of other participating agencies to see if there is a resource match that can satisfy a mutually beneficial exchange of tracking time. At a minimum, a basic planning language could be developed so that ambiguity in searching for exchanges is reduced.

3.3 Common Information Sets

As mentioned previously, one of the major constraints in determining feasible plans are spacecraft viewperiods and major events. If a reliable set of user requirements profiles and associated viewperiods and events can be developed, then a set of software tools, some currently in use and which are under development (see Werntz (1992)), can be used to find feasible swaps among networks. Without a set of common and verified viewperiods for users, the ability to determine the impact of various exchanges on each agency's plans would be very labor intensive.

Algorithms for mission viewperiods by network location and link margin requirements seem to be natural first step in developing a reliable set of resource-mission matches. It is clear that such information is already available for each network alone; extending this effort to the international arena is likely not to be a daunting task.

3.4 Planning Tools

There are several software planning tools that

can assist planners in determining the benefits and impacts of possible reallocations of tracking times. Specifically, there is a PC-based program (see Werntz (1992)) that can add or delete resource availability during any schedule period and determine the average lost time impact on the plan. This utility could provide a first step in locating exchanges of time among agency plans. For example, suppose that during the fourth quarter of 1995 there is high contention on the DSN and if 100 hours of 70M equivalent support could be found for weeks 48-49 contention could be significantly reduced to a more manageable level. In exchange there could be up to 120 hours of 70M time available in weeks 2-5 1996.

3.4 Standardizing Agreements

While the development of common standardized requirements, resource availability, and network plans is necessary to facilitate the determination of feasible exchanges, implementing the exchanges into individuals plans is another matter. Like DBRS we believe that getting involved in international resource conflict resolution is a thorny problem. Instead, within each agency there should be a responsible group that identifies and coordinates potential exchanges, which has the authority to sign a work order type of arrangement to exchange tracking time (or science for tracking time). For the RAP process it should be the responsibility of the RAT to coordinate with TDA to identify potential exchanges and present them to the RARB to be ratified.

4. SUMMARY

The purpose of this paper was to expand on the theme developed by Dumas, Briggs, Reid, and Smith in which they call for the development of standardized agreements and better communication and software to facilitate international cross-support. Considerable effort and progress has been made in improving data standards and communication frequencies among ground data systems. However, not much emphasis has been placed on standardizing and integrating requirements and planning databases among agencies. Without a common set of planning standards or databases it is virtually impossible to identify and

determine the impact of potential exchanges of tracking times among agencies.

The RAP process used at JPL to plan DSN tracking time has matured to the point that it is highly automated. It could be a basic starting point for designing common planning databases. When the development of common/standardized requirements, resource availability, and network plans is resolved, it will become a considerably easier, if not a routine task, to determine feasible exchanges. As a first step we would suggest the following course be pursued:

- Each network's capacity and maintenance requirements, along with a profile of resource availability become part of a common database that can be accessed.
- An international joint committee on network resource planning should be created to develop standard requirements and schedule formats that can be accessed. These would include viewperiod determination and long-range and short-range plan formats.
- Basic software tools should be jointly developed to assist in finding unused times or feasible exchanges that could be acted upon.
- Simple exchanges of tracking time or tracking for science should be advanced through "pre-approved" standard work order arrangements.

5. REFERENCES

- Berner, Carol A., Ralph Durham and Norman Reilly, "Ground Data Systems Resource Allocation Process," *Goddard on Space Applications of Artificial Intelligence*, May 1989.
- Burke, Gene, "Resource Allocation Planning," Report to the Executive Council, Jet Propulsion Laboratory; Flight Projects Office, Multimission Operations Systems Office, October 29, 1992.
- CCSDS Radio Frequency and Modulation Report* ("Green Book") Issue 2, 1990.
- Dumas, Larry N., G. Briggs, M. Reid and J. Smith, "Opportunities for International

Collaboration in Deep Space Communication and Tracking," *AIAA/JPL 2nd International Conference on Solar System Exploration*, August, 1989.

JPL Ground Data Systems Long Range Resource Allocation Plan January 1993-December 2000, Jet Propulsion Laboratory, JPL D-8543, May 1991.

Johnson, Craig and David Werntz, "A New Methodology for Resource Allocation and Planning," paper presented at *Space Conference*, Houston, TX, November, 1987.

Posner, Edward C. and Robertson Stevens, "Deep Space Communications," in *Encyclopedia of Physical Science and Technology*, Vol. 4, Academic Press 1987.

RAP Management Guide, MOSO Mimeo, Jet Propulsion Laboratory, 1992.

"Resource Allocation Review: August 4 1992," Resource Allocation Review Board Presentation, Jet Propulsion Laboratory.

Werntz, David, *FASTER Functional Requirements Document*, D-9962, JPL, Pasadena CA, August 1992.

Appendix A Mission Set

RESOURCE ALLOCATION PLANNING				
User/Mission Planning Set 1993-2000				
Project	Launch	End of Prime Mission	End of Extended Mission	Subnet
ONGOING APPROVED PROJECTS				
DSN	—	—	—	70M/34S/34H/34B
DSS	—	—	—	70M/34S/34H/34B
GSSR	04/01/85	—	—	70M
ICE	08/12/78	08/12/88	12/15/00	70M
PN10	03/02/72	01/15/74	12/30/99	70M
PN11	04/05/73	01/15/75	08/01/95	70M
PN12	05/20/78	01/18/79	10/08/92	70M/34S
RASA	—	—	—	70M
VGR1	09/05/77	12/31/80	12/15/15	70M/34S/34H/34B
VGR2	08/20/77	10/15/88	12/15/15	70M/34S/34H/34B
CAS	10/13/87	TBD	TBD	70M/34H/34B
GLL	10/18/88	10/07/97	12/31/99	70M/34S/34H/34B
ISTP-Geotail	07/14/92	07/14/95	TBD	34S/28M
ISTP-Wind	01/15/94	08/15/98	TBD	34S/28M
MGN	05/04/89	05/04/91	06/21/93	70M/34S/34H
MO	08/25/92	02/03/96	12/22/97	70M/34H/34B
ULS	10/05/90	08/11/95	12/30/00	70M/34S/34H/34B
HRMS	01/01/98	12/30/00	—	70M/34B
MARS-94	03/15/95	08/01/95	TBD	70M/34H
CANDIDATE PROJECTS FOR ADVANCED PLANNING				
ISO	11/94	11/98	N/A	34S/DSS-12/28M
DSPSE	1/21/94	9/94	N/A	70M/34S/28M
PSI/Foer	11/23/94	12/10/97	12/30/98	70M/34H
NEAR	01/15/98	8/30/00	8/30/05	70M/34H
PIUG Flyby	02/01/98	TBD	TBD	34H
MESUR	12/09/98	2/01/06	2/01/08	70M/34H/34B
SOCCEP	06/01/00	TBD	TBD	34B

HRMS - High Resolution Microwave Survey
ISO - Infrared Space Observatory
MESUR - Mars Environmental Survey
NEAR - Near Earth Asteroid Rendezvous
DSPSE - Deep Space Program Science Experiment
SOCCEP - Sample of Comet Comet Earth Return

AN INTELLIGENT PLANNING AND SCHEDULING SYSTEM FOR THE HST SERVICING MISSIONS

N94-23964

Jay Johnson, Lynn Bogovich, Alan Tuchman, Andrew Kispert,
Brenda Page, Christian Burkhardt, Ronald Littlefield and David McLean
Bendix Field Engineering Corporation (Allied-Signal Aerospace Company)
Seabrook, MD 20706, USA.

William Potter, William Ochs and Andrew Dougherty
Goddard Space Flight Center
Greenbelt, MD 20771, USA

ABSTRACT

A new, intelligent planning and scheduling system has been delivered to NASA-Goddard Space Flight Center (GSFC) to provide support for the up-coming Hubble Space Telescope (HST) Servicing Missions. This new system is the Servicing Mission Planning and Replanning Tool (SM/PART). SM/PART is written in C and runs on a UNIX-based workstation (IBM RS/6000) under Motif.

SM/PART effectively automates the complex task of building or rebuilding integrated timelines and command plans which are required by HST Servicing Mission personnel at their consoles during the missions. SM/PART is able to quickly build or rebuild timelines based on information stored in a Knowledge Base (KB) by using an Artificial Intelligence (AI) tool called the Planning And Resource Reasoning (PARR) shell. After a timeline has been built in the batch mode, it can be displayed and edited in an interactive mode with help from the PARR shell. Finally a detailed command plan is generated. The capability to quickly build or rebuild timelines and command plans provides an additional safety factor for the HST, Shuttle and Crew.

Key Words: Hubble Space Telescope, HST Servicing Mission, planning, scheduling, AI, expert system.

1. INTRODUCTION

HST Servicing Missions are Shuttle missions which are expected to occur every three years to upgrade or replace failed HST

components and to help the HST function to its fullest extent over its 15-year mission lifetime. SM/PART helps HST personnel generate "integrated timelines" and "command plans" that describe the activities and detailed procedures to be performed by the HST, Orbiter, Orbiter Crew and ground systems during the HST Servicing Missions. SM/PART has been built upon AI/expert system technology that has evolved through several expert systems delivered to NASA-GSFC (McLean *et al*, [1]).

1.1 The ERBS System

In 1987, the ERBS-TDRSS Contact Planning System (McLean *et al*, [2]) was delivered to the Earth Radiation Budget Satellite (ERBS) Flight Operations Team at NASA-GSFC to generate requests for communications support from the NASA Tracking and Data Relay Satellite System (TDRSS). This system was written in C and implemented on an IBM PC/AT.

The ERBS system is able to use external scheduling environment data and scheduling heuristics to automatically build a schedule of TDRSS requests. After a schedule is built, a graphical timeline is displayed so that users can edit the schedule in an interactive mode, while obtaining "expert" help from the system.

The ERBS system is able to generate a 1-week schedule of TDRSS requests in a few minutes, compared with several hours which were required by the superseded manual method. The ERBS system has been used steadily since its delivery. Also, the ERBS system has been modified and enhanced

several times to meet changing requirements. These changes were easily made because of the AI technology used in the system (McLean [3]).

1.2 The EPPS

In 1991, the Explorer Platform Planning System (EPPS) was delivered to the Extreme Ultra-Violet Explorer (EUVE) Flight Operations Team at NASA-GSFC (McLean *et al*, [4]). The EPPS uses much of the AI/expert system technology from the ERBS system. Like the ERBS system, the EPPS uses a KB to store scheduling heuristics and an AI shell (PARR) to build schedules in both the batch and interactive scheduling modes. As with the ERBS system, the EPPS uses alternative scheduling strategies to perform conflict resolution in addition to more traditional conflict avoidance techniques (McLean *et al*, [5]).

The EPPS provides several enhancements to the ERBS system. First, the EPPS runs on a UNIX-based workstation with X-Windows/Open-Look. Also, the EPPS schedules several types of EUVE mission support activities, in addition to TDRSS service requests which the ERBS system schedules. Finally, the EPPS uses an Ethernet to electronically receive resource data from the Flight Dynamics Facility at NASA-GSFC, receive planning data from EUVE Investigators at the University of California at Berkeley, exchange TDRSS schedule data with the Network Control Center at NASA-GSFC, and send sequences of EUVE command procedures to the Command Management Facility at NASA-GSFC.

1.3 SM/PART Overview

Now, a new, planning and scheduling expert system, SM/PART, has been built and delivered to NASA-GSFC (Johnson *et al*, [6]). SM/PART not only uses AI/expert system technology from the ERBS and EPPS systems but also provides several enhancements. For example, the SM/PART user interface was built with the help of Motif. The "look-and-feel" of this user-

friendly interface follows the OSF Motif Style Guide.

This paper focuses on two aspects of the SM/PART system which allow SM/PART to effectively automate the process of building or rebuilding integrated timelines and command plans for the HST Servicing Missions. These two aspects are strategic planning and tactical planning.

Strategic planning, as used in this paper, refers to the process of identifying and entering the complex heuristics that are used during tactical planning to put events on a timeline. To help SM/PART users develop and capture the strategic planning data, SM/PART provides powerful on-screen editing capabilities for Data Bases (DBs) and KBs. Data set configuration files allow users to define the specific DBs and KBs that are required to build each particular timeline and command plan. More features of SM/PART strategic planning are described in Section 2.

Tactical planning refers to the process of actually placing external scheduling environment data and HST events (activities and comments) on a timeline. For tactical planning, SM/PART uses the PARR shell to build timelines in either an automatic (batch) scheduling mode or an interactive scheduling mode. More features of SM/PART tactical planning are described in Section 3.

Users are able to build/edit HST Servicing Mission integrated timelines in an interactive scheduling mode with "expert" help from PARR. In this interactive scheduling mode, SM/PART provides graphical displays of timelines where objects on the displays are actively linked to specific DBs and KBs. Figure 1 shows a sample of a section of an HST integrated timeline.

Two basic types of HST Servicing Mission scheduling objects displayed on HST integrated timelines are "activities" and "comments." As these objects are edited

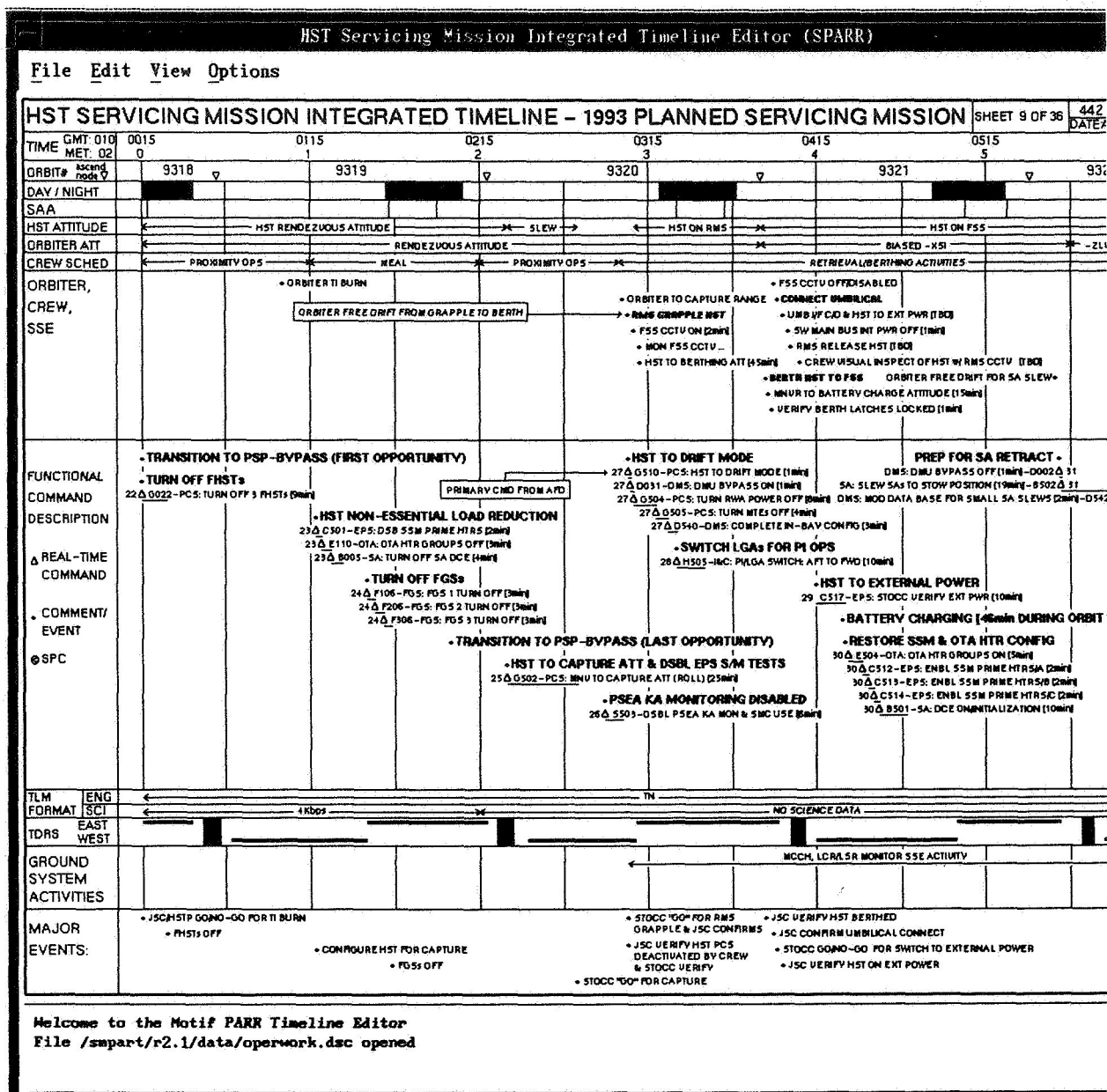


Figure 1. SM/PART Integrated Timeline Section

during an interactive scheduling session, the definitions of these objects in the Event Definition KB are automatically updated.

In addition to HST activities and comments, integrated timelines display several types of scheduling environment data including:

orbit data, orbital events data, HST/Orbiter attitude data, Orbiter/Crew activities, telemetry format data, TDRSS data, ground system activities and major events. Some of this scheduling environment data, such as orbit data, is obtained in electronic form from external sources and cannot be changed

by HST Servicing Mission personnel. Other types of scheduling environment data, such as ground system activities, are entered directly into DBs by HST Servicing Mission personnel and can be changed as needed. Eventually, most of these separate types of scheduling environment data are merged into a single Merged Resources DB before being used to build an Integrated Timeline display.

After a timeline has been built, detailed command procedures (Procs) associated with each scheduled HST timeline activity can be retrieved from a Proc Definition DB and used to automatically build a command plan. A command plan provides the detailed information required by HST Servicing Mission personnel at their operations consoles during an actual mission.

SM/PART also provides users with a hierarchical menu system for overall system control functions and access to UNIX-based utilities for editing ASCII files and performing system and file management functions such as saving, deleting, backing up, restoring and protecting files.

2. STRATEGIC PLANNING

HST activities are defined by users via Activity Event Definition (AD) Forms that contain slots for several types of data items. Some of the items on an AD Form such as "AD number," "AD title," "sequence number," "event type," "display type" and "duration" are relatively simple items. Other items on the AD Form such as "start event," "constraints" and "alternative strategies" are relatively complex. For these complex items, several linked "push-button" and/or "pop-up" menus are provided to make it easier for the users to enter the items. Examples of specific options provided for entering these more complex types of data items are described below.

The "start event" item on the AD Form has four linked data fields with "push-button" or "pop-up" menus so that users may easily

specify the conditions for an activity start time. By selecting menu options, the user may specify that an activity should start when a second (specified) event or resource starts or stops. Also, the user may specify a plus or minus "off-set" for the activity start time relative to the start or stop time of the second event or resource.

The "constraints" item on the AD Form has four linked data fields with "push-button" or "pop-up" menus to allow the user to enter complex constraints or rules for activities. Using these menus, the user can specify that an activity occur when a second specified event or resource occurs or, alternatively, avoid the second event or resource. Also, the user can enter plus or minus off-set times for the various options selected.

The "alternative strategies" item on the AD Form provides various possible alternative scheduling strategies to try, when there is a scheduling conflict. These options include trying to schedule the activity just before or just after a conflicting event, or trying to schedule an activity during the resource window that occurs just prior to or just after the resource window where the scheduling conflict occurs.

HST "comments" represent another major type of HST event that is scheduled. Comments are put into the Events Definition KB via a Comment Event Definition Form that has a structure similar to the Activity Event Definition Form just described.

To make it easier for users to set up sequences of activities for a command plan, SM/PART provides users with a Sequence Definition Form. This form allows the user to define items such as "sequence number," "sequence title," "sequence description," "initial conditions," "duration," "ADs in each sequence," and "special AD ordering instructions."

To make it easier for users to enter the detailed procedures which are required to build command plans, Proc Definition

Forms are provided. These forms allow users to enter procedure information such as: the specific procedures that are to be performed by operations personnel for each AD, the effects of each procedure, the duration of each step/substep, and the actions expected in space and throughout the ground system.

3. TACTICAL PLANNING

To build a timeline, PARR first reads information from the Merged Resources DB and Event Definition KB. The Merged Resources DB provides data that effectively defines the initial scheduling environment while the Event Definition KB provides the event definitions and scheduling heuristics that are used by PARR to place events on the timeline during the scheduling process.

Next, events are placed on the timeline in a batch mode. As each HST event (activity or comment) is considered for placement on the timeline, PARR dynamically allocates an internal frame structure to represent the event and uses the (current) scheduling environment data and the strategic planning data to find a place for the event on the timeline. If there are no conflicts, the event is put on the timeline and the scheduling environment and Event Definition KB are updated. If resources are not available or if constraints are violated, then alternative scheduling strategies are used to try to resolve the conflict. If there is a scheduling conflict that cannot be resolved then a prominent "VIOLATION" message is written in the Event Definition KB.

After a timeline has been built in the automatic scheduling mode, the timeline can be displayed on the terminal screen. This graphical timeline can be edited in an interactive scheduling mode with expert scheduling assistance from PARR. Interactive scheduling and editing are possible because all of the objects on the timeline are closely connected to specific DBs and KBs and PARR is constantly active.

For example, if the user clicks on an activity with the mouse and "drags" it to a new, valid position, then the Event Definition KB is automatically updated. If the activity is dragged to a place where a scheduling constraint is violated then a prominent "VIOLATION" message will be displayed on the screen.

After a satisfactory timeline has been developed, a command plan providing detailed command procedures for each timeline activity can be automatically generated. This command plan can be automatically synchronized with its corresponding timeline if changes in either the command plan or the timeline are made. A graphical "What-You-See-Is-What-You-Get" print of a command plan can be generated for HST personnel to use at their terminals during on-going missions.

4. SUMMARY

In this paper, the AI/expert system capabilities of SM/PART have been discussed from the perspective of strategic planning and tactical planning.

From the strategic planning perspective, complex event definitions and scheduling heuristics may be easily generated and stored in a strategic planning KB. Information in this strategic planning KB is used by PARR to control the automated scheduling processes in both the batch and the interactive scheduling modes.

From the tactical planning perspective, "intelligent" scheduling capabilities are provided by the PARR shell and a graphical integrated timeline. PARR is able to use scheduling environment data from the Merged Resources DB, strategic planning data from a strategic planning (Event Definition) KB and detailed command procedure data from the Proc Definition DB to effectively automate the process of building HST Servicing Mission timelines and command plans.

The capability to quickly build or rebuild HST Servicing Mission timelines and command plans provides a safety factor for the HST, Shuttle and Shuttle Crew in case unexpected events occur during the mission.

ACKNOWLEDGEMENTS

The authors wish to thank Patricia Lightfoot and Dorothy Perkins at NASA-GSFC (Code-510) and Ellen Stolarik and Dave Warren at Bendix Field Engineering for their continual support and many contributions to the IEPS task. This work was supported partially by NASA Contract NAS5-27772.

REFERENCES

1. McLean, D., Tuchman, A., Johnson, J., Yen, W., Page, B., Bogovich, L., Burkhardt, C., Broseghini, T., Klein, S., McIntyre, J., Dao, H., Littlefield, R., and Potter, W., 1992, "Software Reuse in Spacecraft Planning and Scheduling Systems." In *Spaceops 92 - Proceedings of Second International Symposium on Ground Data Systems for Space Mission Operations*, (in press), Nov. 16-20, Pasadena, CA.
2. McLean, David R., Littlefield, Ronald G. and Beyer, David S., 1987, "An Expert System for Scheduling Requests for Communications Links between TDRSS and ERBS." In *Telematics and Informatics*, Vol. 4, No. 4, Pages 253-261, Pergamon Journals Ltd.
3. McLean, David R., 1991, "Maintaining an Expert Planning System: a Software Tools Approach." In *Institutionalizing Expert Systems: a Handbook for Managers*, Pages 91-98, Prentice Hall.
4. McLean, David R., Page, Brenda J. and Potter, William J., 1990, "The Explorer Platform Planning System: an Application of a Resource Reasoning Shell." In *Proceedings of First International Symposium on Ground Data Systems for Spacecraft Control*, Pages 195-200, Darmstadt, FRG, June 26-29.
5. McLean, D., Page, B., Tuchman, A., Kispert, A., Yen, W. and Poter, W., 1992, "Emphasizing Conflict Resolution Versus Conflict Avoidance During Schedule Generation." In *Expert Systems with Applications*, Vol. 5, Pages 441 - 446, Pergamon Press Ltd.
6. Johnson, J., Tuchman, A., McLean, D., Kispert, A., Bogovich, L., Burkhardt, C., Page, B., Littlefield, R., Potter, W., and Ochs, W., 1992, "HST Servicing Mission Planning and Replanning Tool." In *Proceedings of the World Space Congress-43rd Congress of the International Astronautical Federation*, August 28-September 5, Washington, DC.

SCHEDULING THE FUTURE NASA SPACE NETWORK: EXPERIENCES WITH A FLEXIBLE SCHEDULING PROTOTYPE

Nadine Happell¹, Karen L. Moe², Jay Minnix¹

N94-23965

¹ Stanford Telecommunications, Inc.
1761 Business Center Drive
Reston, VA 22090 USA

² NASA Goddard Space Flight Center
Software and Automation Systems Branch, Code 522
Greenbelt, MD 20771 USA

ABSTRACT

NASA's Space Network (SN) provides telecommunications and tracking services to low earth orbiting spacecraft. One proposal for improving resource allocation and automating conflict resolution for the SN is the concept of flexible scheduling (Ref. 1). In this concept, each Payload Operations Control Center (POCC) will possess a Space Network User POCC Interface (SNUPI) to support the development and management of flexible requests. Flexible requests express the flexibility, constraints, and repetitious nature of the user's communications requirements. Flexible scheduling is expected to improve SN resource utilization and user satisfaction, as well as reduce the effort to produce and maintain a schedule. A prototype testbed has been developed to better understand flexible scheduling as it applies to the SN. This testbed consists of a SNUPI workstation, an SN scheduler, and a flexible request language that conveys information between the two systems. All three are being evaluated by operations personnel. Benchmark testing is being conducted on the scheduler to quantify the productivity improvements achieved with flexible requests.

Key Words: Scheduling, flexible request language, space communications, prototyping

1. INTRODUCTION

Providing communications and tracking support to low earth orbit spacecraft has long been one of NASA's primary objectives. NASA's SN, in conjunction with the various ground based communications networks, is designed to provide these necessary support functions.

1.1. The Space Network

The SN is comprised of three main elements: the space segment, the ground segment, and the control segment. All three elements are currently undergoing an evolution to meet user needs in the late 1990s and beyond.

The space segment consists of a constellation of three Tracking and Data Relay Satellite (TDRS) spacecraft that provide forward and return communications and tracking services for customer spacecraft. The future SN may consist of a mixed constellation of TDRS and TDRS II (the next generation TDRS) spacecraft, eventually evolving to a constellation solely of TDRS II spacecraft. While TDRS II is expected to provide some new functionality, the operations concept will be essentially the same as the TDRS operations.

The main element of the ground segment is the White Sands Ground Terminal (WSGT), which provides the communications link between the TDRS spacecraft and the ground. In the near future, a Second TDRS Ground Terminal (STGT) will join WSGT in providing space to ground link support. Over the next several years, both ground systems are planned to receive significant upgrades to address the changing SN environment.

The control segment is responsible for managing SN resources. The primary component of the control segment is the Network Control Center (NCC), which schedules communication events on the SN. In the next few years, the NCC will undergo several upgrades in response to upcoming programs (e.g., STGT). In the late 1990s, the NCC is scheduled for an upgrade, which will significantly enhance the management of SN resources.

1.2 Scheduling the Space Network

Current scheduling practices, which rely heavily on manual conflict resolution strategies, are adequate for the current SN environment. Future changes to the SN are expected to severely impact the capability of the current scheduling operation. These changes could include partial or complete failures of TDRS spacecraft, new services proposed for TDRS II and an anticipated increase in mission loading. Furthermore, the SN has a requirement to support both classified and unclassified customers, therefore, customer visibility into the composite schedule remains severely restricted, complicating the conflict resolution process.

As part of the NCC upgrade, a customer POCC interface called SNUPI is being proposed. One of the primary functions of SNUPI is to support the scheduling interface between the POCC and the NCC. SNUPI and the NCC will implement a concept called flexible scheduling in order to increase scheduling efficiency.

1.3. Space Network User POCC Interface

Current operations use multiple systems to support communication interfaces between user POCCs and the NCC. For example, the User Planning System (UPS) is used for scheduling the SN, while a different interface is used for real-time operations. SNUPI is proposed as a complete interface tool kit that the POCC can access for all of its interactions with the NCC. A key feature of SNUPI is its support of the flexible scheduling concept.

2. FLEXIBLE SCHEDULING

Current SN scheduling schemes involve the submission of individual requests for specific resources at specific times. Flexible scheduling, however, permits POCCs to express requests for communications support in terms of their flexibility, repetition characteristics, and service and event constraints (Ref. 2). Flexibility may be expressed in terms of:

- Service start times (e.g., between 1200 and 1215)
- Service duration (e.g., 15 to 20 minutes long)
- Requested resources (e.g., any SA antenna on any TDRS)
- Repetition characteristics (e.g., repeat once/orbit)
- Service and event constraints (e.g., 2 to 5 minutes after sunlight entry)

Repetition characteristics may include:

- Number of event repetitions (e.g., 10 repetitions)
- Event repetition cycles (e.g., repeat every orbit)
- Summed event duration (e.g., as many repetitions as needed to provide a total of 7 hours of support)
- Repetition restrictions (e.g., 10 minutes minimum gap between repetitions)

Service and event constraints may refer to:

- Orbital activities (e.g., at apogee)
- Calendar activities (e.g., only on weekdays)
- POCC defined activities including other requests (e.g., after request Y starts)

An example of a flexible request embodying these capabilities is a request to schedule a 15 to 20 minute type A event on any single access (SA) antenna any-

time after 1200 hours, except while over the South Atlantic Anomaly (SAA), every day for the next 2 weeks. This request statement expresses flexibility in the start time and duration of the support and in the resource required. It also expresses the repetitive nature of the required support. The repetition instructions in the request allow the POCC to submit one request while receiving multiple scheduled instances of that request (14 in this example). The request statement also expresses constraints limiting the potential times during which this request can be satisfied, based upon orbital activities. Figure 1 illustrates the flexible and repetitive dimensions of the request. Figure 2 provides a sample of how scheduling can be affected by constraint statements.

Increasing Repeatability

<u>Event:</u> Type A <u>Duration:</u> 20 minutes • <u>Repetition:</u> every day <u>Resource:</u> TDRS E SA1 <u>Start Time:</u> 15:12:36 • <u>Period:</u> next 2 weeks	<u>Event:</u> Type A • <u>Duration:</u> 15-20 minutes • <u>Repetition:</u> every day • <u>Resource:</u> any TDRS SA • <u>Start Time:</u> after 12:00:00 • <u>Period:</u> next 2 weeks
<u>Event:</u> Type A <u>Duration:</u> 20 minutes <u>Repetition:</u> once <u>Resource:</u> TDRS E SA1 <u>Start Time:</u> 15:12:36 <u>Period:</u> Tuesday	<u>Event:</u> Type A • <u>Duration:</u> 15-20 minutes <u>Repetition:</u> once • <u>Resource:</u> any TDRS SA • <u>Start Time:</u> after 12:00:00 <u>Period:</u> Tuesday

Increasing Flexibility

Figure 1. Flexible Scheduling Request Example

The use of flexible requests should improve scheduling operations because the information contained in them concerning time and resource flexibility:

- Facilitates automated conflict resolution
- Increases resource utilization (by scheduling more events)
- Reduces number of requests to be submitted (since multiple events can be scheduled from one request)
- Reduces request-response iterations between the NCC and the POCCs
- Reduces the need to coordinate scheduling options verbally (backup events can be specified in case the primary event cannot be scheduled)

Although more options make conflict resolution more complex, algorithmic solutions exist and automated processes have been successfully used (Ref. 3).

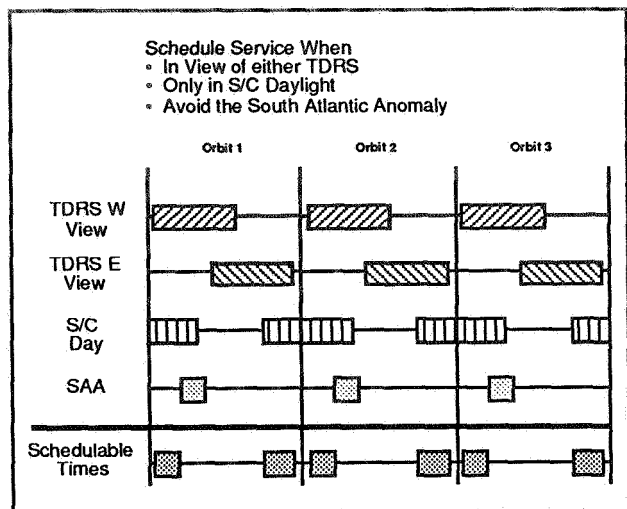


Figure 2. Constraint Relationships Example

3. THE SNUPI PROTOTYPE

The SNUPI prototype illustrates the flexible scheduling concepts as applied to the definition and management of POCC requests. SNUPI runs on a SUN SPARCstation and is written in C. The Transportable Applications Environment (TAE Plus), a NASA developed graphical user interface tool, supported user interface development. The initial version of the prototype was completed in seven man months. This prototype provided access to six top level functions:

- Request generation and editing
- Request validation
- Database management
- Request transmission
- Request status management
- Schedule management

3.1 Request Generation and Editing

3.1.1 The Flexible Request Language

Flexible requests are described to the scheduling system in a flexible request language. The flexible request language used in SNUPI prototype is based on the Flexible Envelope Request Notation (FERN) language (Ref. 4). FERN is English-like and human readable. As used in the SNUPI application, it has a hierarchical structure consisting of definitions of:

- Resources, time intervals, and configuration parameter sets
- Services (or steps)
- Events (or activities)
- Requests (or generics)

Resource definitions consist of the resources that may be scheduled and when they are available. Resource definitions also define pooled or equivalent resources. For example, the resource definition TDRS_East_SA_antenna represents a pooled resource composed of both SA antennas on the TDRS East spacecraft.

Time interval definitions describe relevant time intervals, usually related either to orbital events (e.g., when the spacecraft is in view of a particular relay satellite), or calendar events (e.g., a workday, defined as 09:00 to 17:00 EST Monday through Friday).

Configuration parameter sets indicate how to configure the TDRS spacecraft and ground terminal in order to support a desired service. Typical configuration parameters include data rates, frequency, polarization, and channel identification.

A service definition is comprised of the configuration parameter set and the resources that are required to provide the service. It may also include a list of constraints that apply to that service.

An event definition describes a collection of services as well as the duration of each service. A typical SN event consists of a forward service, a return service and a tracking service. Constraints that apply to the event as a whole, such as "must occur during spacecraft sunlight", are also stated in the event definition.

The request definition describes what events are to be scheduled and how often. Thus, the number of repetitions and the repetition cycles are part of the request definition. Requests also specify what events may be scheduled as backups, if the primary event(s) cannot be scheduled.

3.1.2 The Request Editor

Although the flexible request language is very readable, it still involves keywords and a structured syntax. In order to avoid forcing the POCC operator to learn tedious language syntax, a form fill request editor was developed for the SNUPI prototype. The editor allows the operator to create, edit, or delete the definitions noted above. Each definition is saved and can be retrieved for use in building other definitions. In general, there is a form for each definition, with an associated form for the specification of constraints. The forms support key-in data entry, menu selections, and check boxes. On-line help is provided as well.

The request editor takes the information entered in the

forms and automatically formats it into the flexible request language syntax. At any point in the data entry process, the operator may elect to view the resultant definition in the request language. A text editor is provided for the knowledgeable operator who wishes to work directly in the request language. Syntax and error checking would be performed to the fullest extent possible in an operational system, but only limited error checking was implemented in the prototype.

3.2 Request Validation

Once a flexible request has been defined, the POCC operator should validate it to ensure that it will generate the type, quantity, and approximate placement of scheduled events desired. The request validation facility expands the request into its multiple individual instances and plots these instances on a graphical timeline. Selected constraints may also be plotted. This process allows the operator to see how many request instances will be generated from the request definition and where they are likely to be placed on the schedule. Missing, misplaced, and undesired request instances will be apparent. The operator then may modify the request definitions to correct these problems. Once the operator is satisfied with how the request expands, it is marked as valid.

Figure 3 shows a typical request validation screen. The operator may choose to expand the request along a relative timeline for general validation, or expand the request with respect to a specific time interval. The request may be expanded for one day, one week, or any arbitrary length of time the operator chooses. For example, if the repetition cycle is once a day, the operator may choose to expand the request for a week in order to view a number of repetitions.

3.3 Database Management

Validated definitions for requests and time intervals are entered into a database, and placed under configuration management. Once a definition is entered into the database, modifications to that definition would be tracked and controlled. The database manager provides the operator with the ability to add, delete, view, and sort definitions. The database manager would also support resource inventory updates from the NCC.

When a request is added to the database, the operator is prompted to enter the life span of the request. The life span of a request defines when the scheduling process should start to use the request, and when its use should be terminated. The life of a request may span multiple scheduling periods.

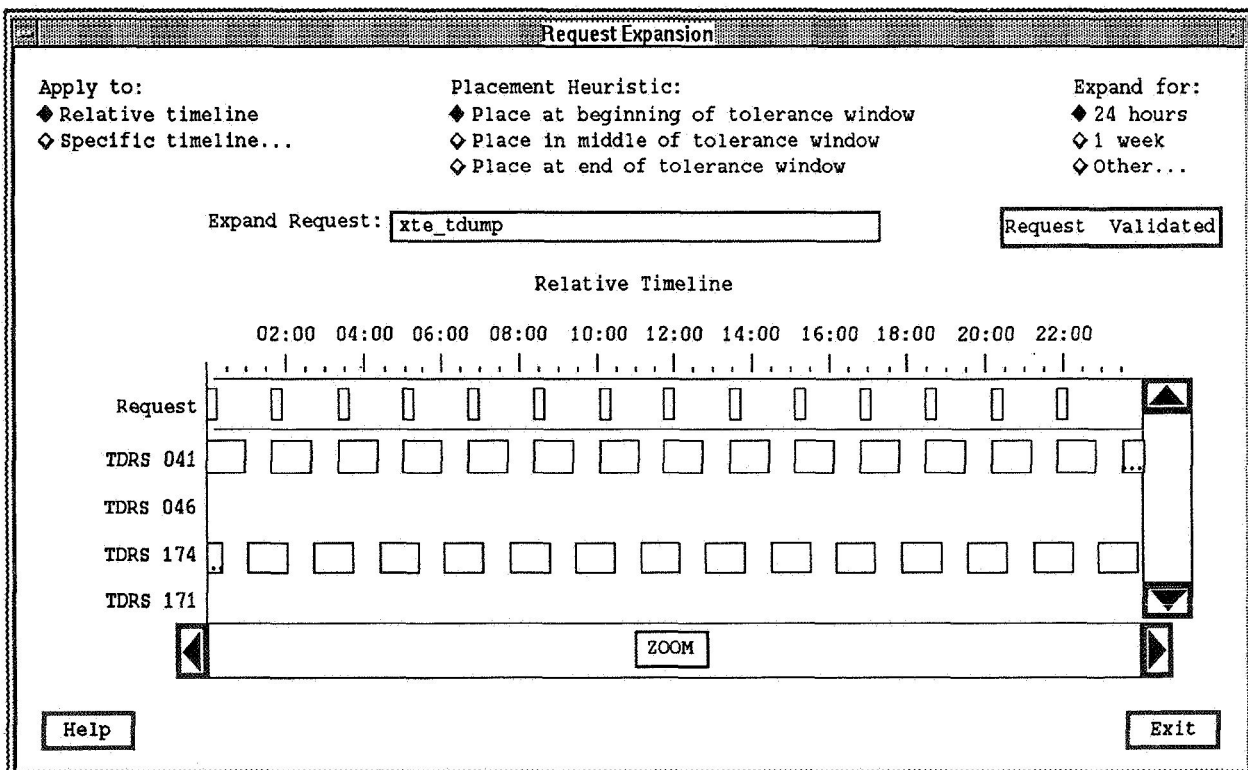


Figure 3. Flexible Request Validation Example

3.4 Request Transmission

The SNUPI prototype is part of a scheduling testbed which includes a flexible scheduling system called the Request-Oriented Scheduling Engine (ROSE) (Ref. 5) which prototypes the NCC scheduler. Requests residing in the SNUPI database can be transmitted via a TCP/IP connection to the scheduling system. When the scheduling system receives a request, an acknowledgment is transmitted back to SNUPI and displayed.

3.5 Request Status Management

SNUPI maintains and modifies the status of all requests from initial request generation, through request transmission, to schedule result. The status display prompts the operator to enter the scheduling period of interest, and then lists all requests relevant to that period and their current status for that period. A request status history is also available.

3.6 Schedule Management

After the NCC scheduling prototype has produced a schedule, each POCC's resulting events are transmitted to their SNUPI. The operator may display these scheduled events on a graphical timeline. Since SNUPI may be managing multiple schedules simultaneously, the operator must specify which scheduling period is of interest, and whether the primary or a contingency schedule is to be displayed. The operator may choose from a variety of display grouping options, such as group by resource scheduled. Details on any particular event may be viewed. The schedule may also be displayed in a tabular form. Information concerning unscheduled events and schedule summary statistics (e.g., total minutes allocated) is available as well.

4. SNUPI EVALUATION

The SNUPI prototype user interface and the Flexible Schedule Request (FSR) concept were evaluated by collecting subjective comments from potential users. The objectives of the evaluation were to determine the acceptability of the FSR concept from the mission operations perspective, and to gather feedback on the techniques used in the SNUPI prototype user interface. The evaluation plan addressed the process to be used for each evaluation session, identification of participants and evaluation materials.

Evaluations were generally conducted with one participant at a time and were completed in about an hour. For each evaluation session, participants were provided

with a brief overview of the SNUPI prototype, the FSR concept and the objectives of the evaluation. The SNUPI prototype was then demonstrated, following a strict scenario which highlighted features of the FSR concept (as described in Section 3), proposed capabilities to support the concept, and several user interaction techniques. Finally, reactions were obtained by having participants respond to an interview questionnaire.

Over fifty people participated in the SNUPI prototype evaluation. Participants were members of Goddard mission operations teams, managers and developers of operational systems. The evaluation material included a handout of the SNUPI overview, a brief description of the FSR concept, and copies of major SNUPI prototype display panels. The questionnaire requested information about the evaluator's background to determine their personal experience with systems similar to SNUPI in function, and asked whether they considered themselves to be managers, developers or operators. Statements were worded positively and then scored by evaluators as 5 for "strongly agree", 3 for "neutral" (or don't know) and 1 for "absolutely do not agree". The statements were intended to elicit subjective feedback regarding the perceived utility of the FSR concept. Words like "useful" were used when exploring the proposed operations concept, and "clear and easy to understand" for human-computer-interface concepts. Participants were also asked to comment on overall prototype strengths and weaknesses.

Responses to the questionnaires were numerically scored and then analyzed to calculate the average response to each statement by participant category and in total. Responses that were rated both high and low (20th percentiles) were further examined, with correlating comments from the participant groups.

The overall result of the evaluation indicated strong approval of the FSR concept, with scores ranging from 3.07 to 4.81, and averaging 4.21. In general, graphic displays of schedules were well received. Low rated statements tended to point to human-computer interface (HCI) difficulties. It was recognized that while requesting scheduled services is a complex task, SNUPI tools need to be more intuitive, perhaps prompting for the next required operator input. Participants recommended that future scenarios deal with some of the complexities currently experienced with SN scheduling, including handling rejected requests, handling more than a single request, and making modifications to scheduled events. For example, an infrequent activity may require that a single instance of a routine request have expanded service support.

The evaluation also indicated that users anticipate that extensive training would be required to develop detailed knowledge of the FERN language. Another key evaluation result was that scheduling SN resources is only part of the planning function for each GODDARD mission. Participants recommended integrating SNUPI scheduling functions with other POCC mission planning activities and tools.

5. CONCLUSIONS AND FUTURE EFFORTS

A key benefit of the flexible scheduling request concept is the shift of a significant conflict resolution effort from humans to computers. Whereas the current space network scheduling approach requires POCC operators to identify specific resource and time requirements in a fixed format request, the flexible scheduling request concept allows customers to describe their requirements in a readable language which accommodates flexibility. The FSR operations concept minimizes request-response iterations between the network scheduling system and the customer since multiple events can be scheduled from a single request (using repeatability specifications). Also, backup events can be identified and substituted in case the primary service is not available. The FSR concept supports automated conflict resolution strategies, since tolerances in start times and duration are provided. More events are scheduled, thus supporting effective TDRS resource utilization. The time to generate a week's worth of schedules can be reduced to hours instead of days. Finally, the potential exists to evolve the FSR concept into a standard for exchanging scheduling information on resource requirements and constraints with international space networks.

Currently a second version of the prototype is in process to address some of the weaknesses identified by the evaluators. One particular improvement is the addition of a new display illustrating the relationship of different services within an event. This display is included in the request editor, and later may be made available to both the request validation and schedule management facilities. A new graphical format is also planned to allow easy definition of time intervals.

To address the need for integrated tools in mission planning and SN scheduling which was identified by the evaluators, the prototyping effort will also explore ways to automate the link between mission planning tasks and the generation of flexible scheduling requests for SN resources. This prototyping effort will integrate SNUPI with an operational mission planning tool called the Explorer Platform Planning System (Ref. 6).

Smart HCI capabilities could be incorporated within the SNUPI prototype. Context sensitive defaults and intelligent error checking could be provided. For example, SNUPI may recognize that the majority of events defined in its database include a tracking service, and query the operator for the inclusion of a tracking service in any new event definition. The interaction style also could be extended to a demonstrational interface (Ref. 7). Demonstrational interfaces go beyond direct manipulation and allow the operator to abstract and generalize actions performed for repetition on similar objects. An intelligent demonstrational SNUPI would allow direct manipulation of event and service icons on a timeline in order to describe a request. Using heuristics, SNUPI then would determine the constraining factors and translate this information into the request language.

REFERENCES

1. Moe, Karen L., ed. Dec. 1990. Space Network Control Conference on Resource Allocation Concepts and Approaches, Conference Proc. NASA CP-3124.
2. Stanford Telecommunications, Inc. Jan. 21, 1992. Space Network Control System/Payload Operations Control Center Flexible Scheduling Concepts, Tech. Rept. TR-91-108, GSFC Contract NAS5-31260.
3. Munns, S., N. Goodman, T. Welden, and D. Crehan. Aug. 1991. Lessons Learned Prototyping Flexible Scheduling for the TDRSS Domain, CSC, Loral AeroSys and NASA/GSFC, DSTL-91-003.
4. Tong, G.M. Sept. 1991. Flexible Envelope Request Notation (FERN) User's Guide, NASA/GSFC, DSTL-91-001.
5. Grasso, B. and G. M. Tong. Jan. 1992. Request Oriented Scheduling Engine (ROSE) Software Maintenance Document, Loral AeroSys and NASA/GSFC, DSTL-92-025.
6. McLean, D., B. Page and W. Potter. 1990. The Explorer Platform Planning Systems: An Application of a Resource Reasoning Shell. In Proceedings of the First International Symposium on Ground Data Systems for Spacecraft Control. Darmstadt, Germany: ESA pp. 195-200.
7. Myers, Brad A. 1992. Demonstrational Interfaces: A step beyond Direct Manipulation. In *Computer*, 25, 8: 61-73.

N94-23966

SOFTWARE REUSE IN SPACECRAFT PLANNING AND SCHEDULING SYSTEMS

David McLean, Alan Tuchman, Todd Broseghini, Wen Yen, Brenda Page, Jay Johnson,
Lynn Bogovich, Chris Burkhardt, James McIntyre, Scott Klein, Hung Dao and Ronald Littlefield

Bendix Field Engineering Corporation (Allied-Signal Aerospace Co.)
Aerospace Building, Suite 500
10210 Greenbelt Road, Seabrook, MD 20706, USA

William Potter

Spacecraft Control Programs Branch
Goddard Space Flight Center
Greenbelt, MD 20771, USA

ABSTRACT

The use of a software toolkit and development methodology that supports software reuse is described. The toolkit includes source-code-level library modules and stand-alone tools which support such tasks as data reformatting and report generation, simple relational database applications, user interfaces, tactical planning, strategic planning and documentation. The current toolkit is written in C and supports applications that run on IBM-PCs under DOS and UNIX-based workstations under OpenLook and Motif. The toolkit is fully integrated for building scheduling systems that reuse AI-knowledge base technology. A typical scheduling scenario and three examples of applications that utilize the reuse toolkit will be briefly described. In addition to the tools themselves, a description of the software evolution and reuse methodology that was used is presented.

Key Words: Scheduling, planning, AI technology, C-based software, software reuse

1. INTRODUCTION

The Spacecraft Control Programs Branch/Code 514 at NASA's Goddard Space Flight Center, with support from Bendix Field Engineering Corporation (BFEC), has been building spacecraft planning and scheduling systems for ground support operations since 1985. Although this technology is Artificial Intelligence (AI) based, it is also C-based because of the

requirement for portability and high performance (McLean, et al., [1]).

Three main applications developed to date are: the Earth Radiation Budget Satellite (ERBS)-Tracking and Data Relay Satellite System (TDRSS) Contact Planning System (CPS) (McLean, et al, [2]), the Explorer Platform Planning System (EPPS) (McLean, et al., [3]), and the Hubble Space Telescope Servicing Mission Planning and Replanning Tool (SM/PART) (Johnson, et al., [4]). The ERBS system runs on an IBM AT, while the EPPS and SM/PART systems run on UNIX-based workstations under Open Look and Motif.

The AI technology used in these applications includes the use of a tactical planning tool called the Planning And Resource Reasoning (PARR) shell. PARR is a heuristic based scheduler which uses frame-like data structures to represent scheduling knowledge. PARR differs from most other scheduling shells in that it places a great deal of emphasis on conflict resolution versus conflict avoidance (McLean et al., [5]).

Before PARR can be used, activity classes must be defined and an external scheduling environmental data timeline containing potential resources and constraints must be available. In addition to PARR, a strategic planning tool is available that allows the user to specify classes of activities in terms of constraints, resources and conflict resolution strategies and then to store these activity classes in a Knowledge Base (KB).

For satellite scheduling, a scheduling environmental data timeline is generated by the Flight Dynamics Facility (FDF) at Goddard. When the activity KB and the environmental data timeline become available, the user can invoke PARR to autonomously generate a conflict-free schedule. Later, the user can invoke an interactive version of PARR to browse and edit the schedule.

The task of building satellite scheduling systems at Goddard has led to the reuse and evolution of earlier versions of the software. This effort has resulted in a development methodology along with new software tools for managing software reuse and evolution.

2. TYPICAL SCHEDULING SCENARIO, REQUIREMENTS AND PROBLEMS

2.1 A Typical Scenario

The scheduling scenario given here is based on satellite scheduling tasks at Goddard. It is a generalization drawn from interviews with several Flight Operations Team (FOT) schedulers.

1. Get external scheduling environmental data from FDF.
2. Define classes of activities to be scheduled.
3. Put external scheduling environmental data on timeline.
4. Put activities on timeline in predefined order.
5. Perform resource allocation, constraint checking and conflict resolution, as required.
6. Generate tentative schedule.
7. Interactively refine schedule until deadline.
8. Expand final schedule into detailed command schedule.
9. Send final schedule to the Command Management System (CMS) for load generation.
10. Keep careful records of what was done.

2.2 Software Requirements

The software requirements for each stage of the scenario include:

1. Establish Ethernet link to the source of external environmental data. Reformat external data to internal scheduling formats.

2. Provide a tool which takes an activity class definition and generates a KB entry for each activity class.
3. Provide a tool which places the external environmental data on a timeline.
4. Provide a tool which places the activities specified by each activity class on the timeline.
5. Perform automatic resource allocation, conflict checking and heuristic conflict resolution.
6. Write the tentative schedule to a file and print a report.
7. Provide graphic timeline browse and edit capabilities.
8. Provide a tool which allows the user to specify sequences of low-level command activities that represent the details of the higher level activities. Provide a tool which expands the higher level activity schedule into the lower level command sequence schedule.
9. Reformat the internal schedule to the external schedule format and establish the Ethernet link to the CMS.
10. Provide a user interface and generate reports so that each of these steps can be easily accomplished and carefully managed.

2.3 Scheduling Problems

At the heart of the scheduling scenario are four basic types of problems that humans encounter in placing spacecraft activities on a schedule:

1. Order of activity placement
2. External Constraints
3. Spacecraft Resources
4. Conflict resolution

When a schedule becomes crowded with many activities it becomes more and more unlikely that new activities will have enough resources available, so that the order of activity placement becomes an important conflict avoidance heuristic. A typical response to this problem is to place the most constrained activities on the timeline first.

Scheduling and planning problems involve predicting external conditions and using the constraint information to decide where activities should be put on a schedule. From the scheduling point of view, external events may be viewed as constraints or resources. As spacecraft activities are placed on a timeline, they are initially constrained by the predicted

orbital events (initial external constraints) and later by other activities already placed on the schedule (current external constraints).

Some types of activities may require the use of sharable or consumable spacecraft resources such as power or a tape recorder. Models of the behavior of these resources are required in order to reflect the changes to the scheduling environment as activities of this type are placed on the timeline.

When conflicts are found due to constraint violations or resource unavailability, heuristics may be used to resolve these conflicts. For example, a conflicting activity may be scheduled at a later time or may use an alternative resource.

The IEPS software toolkit used to solve these problems allows the human scheduler to specify four types of scheduling heuristics corresponding to the four problem types described. A strategic planning tool that allows the user to specify each of these types of heuristics and store them in a the strategic planning KB is provided.

3. EVOLUTION OF THE TOOLKIT

3.1 ERBS System

The purpose of the ERBS scheduling system is to aid the user in building a confirmed TDRSS contact schedule. This is done by specifying the desired times for contacts and sending these requests to the Network Control Center (NCC). Requests for TDRSS time may have to be resubmitted several times when conflicts occur. This iteration with the NCC for confirmed TDRSS time is often a very time consuming task.

The IEPS toolkit started by implementing only the ERBS system requirements, which are a subset of all the requirements in the scenario described above. For example, the Ethernet connection was never available, so a tape drive was provided to obtain the external scheduling environment data file from the FDF. Also, the product of the ERBS scheduling system (a set of reports) had to be hand carried to the mission planning terminal. The ERBS KB building activity was done as a traditional knowledge

acquisition task by a software/knowledge engineer. Finally, ERBS did not require the expansion of the activities into more detailed command sequences because that task was done in the CMS.

After delivering the ERBS scheduling system, the IEPS toolkit consisted of:

1. A set of reusable C modules.
2. A reformatter for the external environmental data timeline.
3. A batch/interactive tactical planning shell (PARR).
4. A set of report generators.

During the maintenance phase of the ERBS system, the toolkit was reengineered so that the system was more maintainable. This reengineering effort included a greater separation of the user interface code from the rest of the system code.

Anticipating future work, a copy of the ERBS software was ported to UNIX-based workstations where the user interface software was reengineered to interface with X-Windows and the reformatting and report generation tools were enhanced and made more generic.

3.2 EPPS

When the EPPS project started, the IEPS team was prepared to reuse a good portion of the ERBS system. For EPPS, PARR was broken down into several more specialized scheduling tools: interactive, batch and merge. Also, a two-tape-recorder model was added to PARR's resource modeling capabilities.

An additional requirement for EPPS was the capability to define sequences of activities that represent the details of each higher level (traditional PARR) activity. Also, there was a requirement to expand the PARR schedule file into time-instantiated sequences. The definition capability was accomplished by use of the database tools and the expansion was done by a special stand-alone tool.

A major development effort for EPPS was to build a KB editor. Early versions of the KB editor utilized the IEPS toolkit user interface tools. Later versions of the KB editor also

utilized the IEPS relational database toolkit. EPPS currently uses the Open Look user interface library, but future applications will be using Motif.

EPPS has Ethernet connections to the CMS to receive external environmental data files and science schedules. The final schedule produced by EPPS is sent to the CMS via this Ethernet connection. Because of the variety of different data types available electronically, the IEPS reformatting toolkit was greatly expanded.

A separate power model was developed for EPPS. This is a sophisticated simulator that is utilized by the FOT to keep track of the health and safety status of the batteries. At the current time, this power model is not an integral part of PARR. However, there are plans to merge the power model with the interactive version of PARR so that the FOT can see the effects of interactive additions or deletions of activities on the health and safety status of the batteries. There are plans to build a simulation toolkit which would support all of the (currently) separate spacecraft simulation efforts.

Documentation tools have also evolved, and these tools have saved many hours of time when documentation for deliveries are required. Documentation tools are used extensively to generate maintenance documents.

By the time of Explorer Platform launch (spring 1992), the IEPS toolkit included the following (NASA-GSFC [6]):

1. An extended set of C modules in various libraries.
2. A set of reformatting/report-generation tools.
3. A set of user interface tools.
4. A set of database tools.
5. An activity class definition tool and a command activity definition tool.
6. A set of batch tactical scheduling tools.
7. An interactive tactical scheduling tool.
8. A set of documentation tools.

3.3 SM/PART

The Hubble Space Telescope (HST) in-orbit servicing activities require detailed planning of the preferred and alternative activities. In addition to these activities, "command plans" are

also required to describe the detailed commands that are to be sent during the mission.

The HST servicing mission scheduling needs are quite a bit different from the other two missions described. One of the differences is that the HST activities are very specialized--the concept of activity class does not fit most of these uniquely defined activities. Nevertheless, virtually every category of the IEPS toolkit contributed to and was enhanced by the SM/PART effort. For example, the database and KB editor tools were enhanced and the code used to interface with Open Look, used by EPPS, was converted to Motif. Also, new user interface displays were created to include different types of timeline displays, PARR's scheduling algorithm was modified to support the more demanding temporal dependency links between activities and at the module level, the IEPS toolkit was extended to include more date/time reformatters.

4. USING THE TOOLKIT TO BUILD SCHEDULING APPLICATIONS

The first step in building new scheduling applications is to configure the reformatting tools and user interface files to specify the domain of activities in terms of forms and menus. Then, script files (shell scripts) are created to link the various tools into a unified system. This unified system is then iteratively refined by reconfiguring and extending the capabilities of the tools as well as by developing new tools until the system meets all the initial and discovered requirements. Rapid prototyping is often used during the initial stages of building new applications in order to enhance the requirements discovery process.

The software engineer starting a new scheduling application would most likely examine the basic requirements and then select the current application that meets most of these requirements. Those parts of the system scripts and menu options which are not required would be eliminated. In addition, the reformatting tools would be reconfigured so that the external scheduling environmental data could be read by PARR.

Next, the names of the external environmental data, in the activity specification tool's menu file,

would be changed to reflect the new application. Any other required changes to these menu files would also be made at this time, until the user is satisfied that the system can adequately specify the types of activities required.

At this point, if a test external environmental data file is available, the user can start using PARR to generate test schedules from the prototype activity classes defined. During this process, the user learns how to use the system and discovers the types of activities that produce the most desirable schedule. This process may go on for some time, but it will eventually establish the basic activity classes required.

Finally, the output schedule is reformatted according to the requirements of the user, and TCP/IP tools may be used to receive resource files from an external source and send the final schedule to an external user. Various reports which allow the user to manage the entire scheduling task will also be prototyped via the report generation tools.

5. MANAGING SOFTWARE EVOLUTION

5.1 Evolutionary Prototyping versus the Waterfall Model

Evolutionary prototyping is a software development methodology that combines the advantages of software reuse and prototyping. Because evolutionary prototyping is based on software evolution, special care must be taken to provide a management structure which supports it (Arthur, L., [7]). The traditional waterfall model does not accommodate the sort of evolutionary development made possible by rapid prototyping capabilities. Further, the "manufacturing model" of software development makes the unrealistic assumption that software can be totally pre-specified as with a mass-produced item.

On the other hand, the evolutionary prototyping approach makes the assumption that software can never be totally pre-specified because the development process is, by nature, a discovery process which results in a continuously evolving specification. Thus, evolutionary prototyping provides a methodology for requirements

discovery through software reuse and prototyping.

Lowel Arthur contends that the term "rapid prototyping" is regarded with suspicion by many customers because many of the traditional prototyping efforts were done in labs away from the potential users with hardware and software that was far removed from the target platform. The result of this type of "throw-away" prototyping effort often led to systems that had little relevance to what the customers could use and were very costly to completely reengineer.

By contrast, evolutionary prototypes are developed with reuse in mind and are built on a stable, reuse library. They are developed in the same language and on platforms similar to the target system. Typically reuse tools are small, well-integrated, and easy to understand and maintain. Reuse tools should also be generic enough to be of use in a wide variety of applications. The rapid, evolutionary prototyping paradigm is a plan-do-check-act cycle which is repeated until all the requirements are met, at which time the system is delivered.

Some of the advantages of evolutionary prototyping are:

1. It is simple to do. Prototyping means concentrating on the essentials of the problem.
2. It provides value in a short time frame with early, easy wins that establish customer commitment.
3. It allows engineers and customers to learn as they go.
4. Often, until something can be demonstrated, the customer will not be able to clearly articulate the needs for the system.
5. It lowers risk because the project can be canceled at any time. When new versions of the prototypes are delivered, managers and customers know the exact state of the project.
6. It increases the ability to deliver bug-free systems with the desired functionality.

A potential management problem with reuse is that traditional productivity measures, such as lines of code, do not reflect the effort and quality of this process. Further, software reuse requires an initial training phase for the software engineers. Therefore, these factors need to be

taken into account when predicting the software effort for the evolutionary prototyping approach.

5.2 IEPS Activities and Software Reuse

The IEPS group consists of software engineers who spend some of their time keeping pace with cutting-edge software technologies. In addition to developing new applications, they look for new technologies that promise to be of value to mission operations systems at Goddard. These activities include:

1. Periodic reviews of new technologies as presented at various software conferences and in journals.
2. Emphasis on software reuse of independent but well integrated tools.
3. Use of rapid prototyping via evolutionary refinement of existing tools.
4. Providing demonstrations of prototypes to the Goddard community for technology transfer and feedback.

Today, the applications initiated by this group are often developed in conjunction with software engineers from other groups. This process results in the transfer of new development technologies to other software engineers.

6. CONCLUSIONS

Software evolution and reuse, given the right environment, works very well. A "life cycle" management structure which allows and encourages evolutionary prototyping techniques is a major requirement for this type of software development. We have demonstrated that the techniques described here can be used to successfully build a satellite scheduling system (EPPS) at nearly half the estimated cost of a traditionally built system. The EPPS was so successful that the development team was awarded a Goddard Productivity Improvement and Quality Enhancement award. Careful management of the toolkit will provide even more payoff for future missions.

7. ACKNOWLEDGEMENTS

The authors would like to thank Patricia Lightfoot at NASA-GSFC/Code 514 and Ellen Stolarik and David Warren at Bendix Field

Engineering Corporation for their continual support and many contributions to the IEPS task. The IEPS task is supported by NASA contract #NAS5-27772.

6. REFERENCES

1. McLean, D., Tuchman, A., and Potter, W., "Using C to Build a Satellite Scheduling System: Examples from the Explorer Platform Planning System," *Telematics and Informatics*, Vol. 8, No. 4, pages 297-312, Pergamon Press, 1991.
2. McLean, D., Littlefield, R., and Beyer, D., "An Expert System for Scheduling Requests for Communications Links between TDRSS and ERBS," *Telematics and Informatics*, Vol. 4, No. 4, pages 253-261, Pergamon Press, 1987.
3. McLean, D., Page, B., and Potter, W., "The Explorer Platform Planning System: An Application of a Resource Reasoning Planning Shell," *Proceedings of the First International Symposium on Ground Data Systems for Spacecraft Control*, Darmstadt, Germany, pages 195-200, June 26-29, 1990.
4. Johnson, J., Tuchman, A., McLean, D., Kispert, A., Bogovich, L., Burkhardt, C., Page, B., Littlefield, R., Potter, W., and Ochs, W., "HST Servicing Mission Planning and Replanning Tool," *World Space Congress*, Washington, DC, August 28-September 5, 1992.
5. McLean, D., Page, B., Tuchman, A., Kispert, A., Yen, W., and Potter, W., "Emphasizing Conflict Resolution versus Conflict Avoidance during Schedule Generation," *Expert Systems With Applications*, Vol. 5, pp. 441-446, Pergamon Press, 1992.
6. NASA-GSFC/Code 514, *IEPS Software Toolkit*, Vol. 1-7, August 1992.
7. Arthur, L., *Rapid Evolutionary Development Requirements, Prototyping & Software Creation*, John Wiley & Sons, 1992.

N94-23967

PLANNING: MANAGEMENT OF PREDICTABILITY AND UNCERTAINTY AND KEEPING ABREAST OF DEVELOPMENTS

Christophe BASTIEN-THIRY

Centre National
d'Etudes Spatiales

18, Avenue Edouard Belin
31055 Toulouse Cedex, France
Tél.: 61.27.38.53 - Fax: 61.28.26.03

Gérard VERFAILLIE

Centre d'Etudes et
de Recherches de Toulouse

2, Avenue Edouard Belin
31055 Toulouse Cedex, France
Tél.: 61.55.71.95 - Fax: 61.55.71.32

ABSTRACT

The purpose of this study is to propose method to set up and control of a space mission plan such as that of the HERMES spaceplane. The interest of this subject, other than its complexity, is due to the need to manage imprecision and uncertainty during a mission, as well as changes in between missions.

Under these conditions, the set up and control of a flight plan require certain special attention and this has led us to define a certain number of qualities:

- * Mastery of complexity in order to resolve conflicts between activities : configuration, resource and time management,
- * Consideration of various criteria such as risk minimization or the attainment of mission objectives.
- * Robustness and flexibility to allow for hazards and deviations from the norm during operation without having to draw up new plans.
- * Aptness for replanning by making changes to the plan without having to set up the whole plan again.
- * Memorization and explanation facility in order to manage developments between missions.

Key Words: scheduling, robustness, flexibility, revision, explanation, design process.

1. MOTIVATION

The study for which we present certain results here is being carried out mainly by the CERT under CNES's Research and Technology activity. As the study is not yet finished, it will be first results that are presented in this article.

Planning and scheduling are traditional, reputedly difficult subjects of study which concern a much

wider community than simply those involved in the space field. Moreover, many tools and methods have already been put forward as an effective response to some of these problems. So two questions come to mind: what specific constraints are imposed by our space activity, and what justifies the need for progress in this domain?

Space activity, as we know, is expensive and uses ever more complicated technological means. This obliges all project designers to consider not only the feasibility and safety aspects but also the optimization of resources as an absolute necessity. What is more, the urgency of this necessity leads to higher order reflection. Tomorrow's requirement will no longer be "to design something technically and economically feasible" (with optimization being taken into consideration afterwards) but to "think of another way designing things so as to master complex systems (in technical, human and economic terms)". In this perspective, the optimization aspects must be considered as a point of view intervening from the very start of the design process in the same way as techniques and safety.

Our thoughts and the studies performed on planning and scheduling tools and methods suggest elements of a solution to this problem. We would be justified in saying that not only do these tools obviously enable a satellite's resources to be optimized after the event but that the methods also assist the design process for space systems. Indeed, the planning and scheduling methods (and also, for example, the diagnostic tools and methods) should influence design choices: how can a system be designed without considering the means of exploiting and controlling it?

2. PROBLEMS POSED IN THE FRAMEWORK OF A SPACE MISSION PLAN

The aim of the work we present falls within the perspectives mentioned above. However, from a pragmatic point of view, we preferred to start with a concrete study corresponding to an actual need. During the preliminary definitions of the study in 1990, the application that appeared the most propitious support for this research was the setting up and control of the flight plan for a Hermes mission.

Although the study is influenced by the problem of the Hermes flight plan, the study is being carried out with the objective of making it generic and reusable for other space missions, and, as we have seen, in the perspective of the general set of problems concerning the design process.

The management of activities during a space shuttle flight is fully planned in advance and updated as the flight progresses (Ref.1). The planning of the Hermes mission (definition of the activities to be performed to attain mission objectives) appeared to fall mainly within the scope of human thought rather than being suitable for automation. Once the activities capable of meeting the mission objectives have been defined, the problem is to schedule them taking account of constraints connected with time and resource utilization: constraints for performing the activities as a whole, constraints with use of systems, time constraints on and between activities and events, constraints connected with the cyclic nature of certain activities, constraints on the use of consumable or renewable resources. The activity model used for the scheduling contains imprecisions (duration and quantity of resources actually needed for the activities) and uncertainties (possibility of failures or need for new activities). So to avoid having to revise the plan too often at scheduling level, we propose a double approach: produce a **robust plan** and execute it with **flexibility**.

The problem is **dynamic**:

- when the flight plan is being set up, the user may want to add or remove certain constraints to improve a given aspect;
- during the flight, because of the drifts and hazards not totally absorbed by the robustness or flexibility of the plan;
- between two flights because of changes in the system and mission, which must result in

corresponding changes in the plan.

The dynamicity to be efficient involve revision, i.e find a new solution plan from the former one. From an operational point of view, it is easy to see that the aim is to limit the alterations to be made to the plan as far as possible, whence the need for a plan revision mechanism that can work on local modifications.

Finally, the plan resolution system must offer means of **explanation** so that those working on the mission in the plan elaboration or control phases have ways of interacting with the solution-seeking mechanism. They must know the origin of a situation, the consequences of a choice or a constraint and be able to reconsider choices or constraints.

3. FOR UNPREDICTABILITY AND UNCERTAINTY: ROBUSTNESS AND FLEXIBILITY

Since it is impossible to set up a global model of the world, it will always remain drifts and hazards during the flight execution; so, to reduce their effects, the chosen model must foresee them. It is why we have introduced the complementary concepts of the robust plan and the flexible plan in order to limit the scheduling revision (Ref.2).

3.1 Robust plan

We consider here only a few of the possible definitions of a robust plan, each of which is connected with a different operational concept. A robust plan scheduling can be defined as a scheduling that reduces to a minimum:

- the maximum loss of attainment of objectives. This definition favours the stability of the objective attainment level.
- the probability of loss of attainment of objectives.
- the maximum change that it can be necessary to make to the scheduling to keep the same level of satisfaction of objectives. This definition favours the stability of scheduling.

These definitions bring in the notion of attainment of objectives, which can be defined from the degree of importance attached to each objective. The last brings in the idea of measure of change of a scheduling, an idea that must be given a sense from an operational standpoint. It is also necessary to have access to information relative to the probability that foreseeable events will occur.

3.2 Flexible plan

Unlike the previous approach, which makes choices but tries to make them as resistant as possible to the perturbations that will inevitably occur, the approach that we designate as flexible limits the choices to the parts of the scheduling that can be based on sufficiently precise and certain models of activities. For the parts where the models are imprecise or uncertain, the choices (like the precise start dates of activities) are delayed. These choices will be made either at the instant when the start of execution has sufficiently restricted the areas of imprecision and uncertainty or, failing this, at the time when the execution forces the choice to be made.

It must be said that these definitions and approaches that attenuate the problem of revising a scheduling are independent of the mechanism chosen for finding the solution and should rather be considered at the time when the mission, its objectives and the corresponding actions are designed, i.e. when the planning is done. In other words, it is the modelling of the problem that will ensure robustness and flexibility. It is worth noting in passing that the identification of the principles of a robust, flexible plan, which took place while the principles of a Hermes mission were being drawn up, allowed the concept of vehicle/ground system independence to be clarified (distribution of responsibilities between the vehicle and the ground segment).

4. FOR DYNAMICITY, EXPLANATIONS AND INTERACTIONS: REVISION, LOCAL MODIFICATIONS AND MEMORIZATION

Let us define the problem of scheduling as follows: Consider a set of tasks coming from the planning activity. The problem consists of scheduling them in time and allocating resources to them. These resources may have specific characteristics expressed as constraints on their use: they may be sharable, consumable, renewable. The tasks are generally linked to one another by linear constraints on the start and finish dates for their execution. Finally, some criteria can be expressed with respect to time (finish a task as early as possible) or to the use of resources (use the resource the least called upon).

There are several principles that can be applied to

solve the scheduling problem thus defined. For the case where we are only interested in scheduling with respect to time constraints (no resource allocation: no disjunction), the best known principles are linear programming (Simplex algorithm) and the PERT method (search for the shortest path). In the cases where resource allocation as well as time must be taken into account, the methods envisaged are arborescent research (Backtrack, Branch and Bound algorithm), expert systems approaches (use of heuristics), constraint satisfaction, or random searching.

Before presenting the two solutions produced in this study in order to consider the dynamic and explicative aspects, we shall describe the principles of constraint satisfaction since it is on these principles that both are based.

4.1 Constraint Satisfaction

Our choice was that of **constraint satisfaction**, based on CSP (Constraint Satisfaction Problem) formalism (Refs. 2-3-4-5). What were the reasons for this choice? We recall that the aim of our studies was not so much to put forward new principles for solving the scheduling problem but rather to develop techniques enabling the solution to be found and, above all, offering the capacity to manage the dynamic aspects of the plan and propose possibilities of interaction with the user. Our problem was thus to choose a solution-seeking principle that was as efficient as possible and capable of being enriched with the target functions. Constraint satisfaction is a subject that has come under study fairly recently (latter half of the '80s) and, all in all, has been little explored. This type of programming quickly proved itself to be extremely efficient in solving highly combinatorial problems, which is the main characteristic of scheduling problems. This technique thus offered assured efficiency and a potentially fertile field to be opened up in our sphere of research. Finally, numerous commercially available tools implementing the CSP method have come into being, thus reducing the development effort needed for our work.

CSP formalism can be presented as an extremely general form of problem solving, on the basis of which equally general methods of reasoning have been developed.

A CSP is defined by (Ref.6):

- a finite set V of variables v and

- a finite set C of constraints c .
 - a finite domain D of possible values is associated with each variable v .
- With each constraint c are associated:
- a subset V' of variables of V connected by the constraint and
 - a relation R of any kind that defines the constraint to be respected between the variables of V' .
- The basic problem of a CSP consists of finding, for each variable of V , a value such that all the constraints of C are respected. But other problems may also be set on the basis of this formalism:
- knowing whether or not a solution to the previous problem exists (without necessarily producing one),
 - determining all the solutions or their number,
 - producing a solution having a specific quality (maximizing a criterion, for example).

All these problems are known to be NP-complete, i.e. there is no algorithm of polynomial complexity that solves them. Their complexity increases exponentially with the size of the problem (number of variables, of constraints and their nature, cardinal of the domains associated with the variable). In spite of this NP-complete character, general methods propose interesting solutions. We present below the two most currently used methods.

4.1.1 Enumeration

The basic algorithm of **enumeration** methods (also called Backtrack method) is a simple arborescent research algorithm consisting of:

- determining an order of variables to be assigned,
- assigning the variables one after another, in the previously chosen order, choosing a value in its domain for each one,
- verifying after each assignment that the situation is consistent, i.e. whether the constraints linking the variable that has just been assigned to those previously assigned are respected,

until:

- all the variables have been assigned respecting the constraints or
- an inconsistency is observed, in which case a new value is tried for the current variable; if no further value is possible for this variable, we backtrack to the previous assignment. Of course, this unrefined algorithm does not avoid the combinatorial aspect of the problem and numerous improvements have been made, like the use of heuristics either for the choice of the order of assignment of the variables (e.g. choose the variables intervening in the greatest

number of constraints first) or for the choice of the value to be assigned (Ref.7). Similarly, instead of backtracking to the previous assignment in case of failure, it is possible to avoid going back over assignments that are not involved in the inconsistency observed (Ref.8).

4.1.2 Filtering

The **filtering** method (also called constraint propagation or consistency enforcement (Ref.9)) uses the principle that any CSP can be transformed into an equivalent CSP that is simpler, even easy (a CSP is said to be easy if it can be solved by the Backtrack algorithm without actually backtracking). Two CSP's are said to be equivalent if any solution of one is also a solution of the other. One CSP is simpler than another if the domains of values associated with the variables and the relations associated with the constraints of the first are included in the domains and relations of the second. In practice, the consistency enforcement method is limited to **arc-consistency**. This method relies on the application of a simple procedure between two variables v_i, v_j which suppresses from the domain of v_i any value that makes it impossible to satisfy the binary constraints between v_i and v_j . This procedure is applied to all pairs of variables until no domain of variables remains to be reduced. It can be generalized to n -uplets but at a cost in computing time which is out of all proportion with the expected gain.

However, this method does not guarantee the existence of a solution and does not provide the means of producing one. In consequence, a reasonable method of obtaining a solution consists of alternating enumeration and filtering by arc-consistency (Ref.10).

4.2 First solution - improvement of the "enumeration plus filtering" method.

The principle of this solution is based on an alternance of enumeration (possibly caused by choices made by the user) and filtering. This principle is accompanied by the memorization of information on the origins of the removal of values. The method starts from the following observation: any removal of a value val_i from the domain of a variable v_i has two possible origins:

- the enumeration mechanism, i.e. a choice that excludes this value, or

- the filtering mechanism, i.e. there are no more values in the domain of a variable v_j , linked to v_i by a constraint c_{ij} , that support it.
The absence of support of val_i by a value val_j of the domain of a variable v_j linked to v_i by a constraint c_{ij} also has two possible origins:
 - non-respect of constraint c_{ij} by the pair (val_i, val_j) or
 - the removal of val_j from the domain of v_j .

We therefore suggest that, whenever a value val_i is removed from the domain of a variable v_i , the origin of the removal, enumeration or filtering, should be stored in memory. For the case of filtering, we shall store the origin of the absence of support for this value by another variable v_j , linked to v_i by a constraint. Thus, by recursion, it is possible to find the set of causes for the removal of a value or set of values, which allows us to find the origins of a situation. Similarly, to reconsider choices or constraints that led to a failure, it is of no use to pay attention to the choices or constraints that are not part of the set of causes identified as being responsible for the situation.

It should nevertheless be noted that the explanation we shall be able to give for the removal of a value will only be the first as, once a value has been removed, the mechanisms for solving CSP's do not pay attention to other possible ways of removing it.

This method thus enables explanations to be given. Furthermore it offers the possibility of managing dynamics. In the case of removal of a constraint or a choice, it is possible to put back at present all, and only all, the values of the variables such as this constraint or choice is a cause of their removal. A new attempt at filtering all these values can then be made in order to restart the solving process from a coherent state.

4.3 Second solution - improvement of the enumeration method: a local modification approach

The principle of this method is: when a failure is found with assignment a value to a variable, instead of reconsidering, as in the Backtrack algorithm, the last choice made for the variable previously assigned, which may have nothing to do with the failure, only those previous assignments that are incompatible with the present are reconsidered. For each of these disassigned variables, a new assignment is attempted with no possibility to

modify the assignment of the variable that caused them to be changed. Success is obtained when all the disassigned variables have been able to be assigned again; if one of them cannot find a coherent assignment, the attempt has failed. Another assignments, if they exist, of the first variable then can be attempted. It can be shown that this method is complete (if a solution exists, it will be found).

Management of the dynamic aspect requires no additional measures:

- in cases where a constraint is removed:
 - either the previous problem had a solution: it is necessarily a solution of the new, less constrained problem, or
 - the previous problem had no solution (failure of the attempt to assign a variable): an attempt to solve the new problem may be started from the context (set of assignments) preceding the failure;
- in cases where a constraint is added:
 - either the solution to the previous problem respects the new constraint and there is nothing to be done, or
 - the solution to the previous problem does not respect the new constraint: it is then sufficient to choose one of the variables connected by the constraint, disassign it and attempt a new assignment.

The essential advantage of this method is that it enables a problem to be solved starting from any context and, in case of revision it finds a solution, if one exists, fairly close to the previous one since its principle is based on local modifications.

5. CONCLUSIONS AND PERSPECTIVES

Each of the methods presented answers a part of the problem. The first, although it offers a mechanism for revision and means of explanation, offers no way to revise locally. The second, although it offers an enumeration mechanism working indifferently in construction or revision mode and is able to find a new solution not too far from the former, has none of the filtering mechanisms which have so notably improved the efficiency of the Backtrack algorithm. Hence the search for a mixed method where the enumeration mechanism would be replaced by a mechanism based on local modifications to which could be added a filtering mechanism with storage of the causes of removal. Another way of mixing the methods is to apply them in different phases:

when the plan is being drawn up or adapted to a new mission, the first method is used, which allows interaction with the people involved; in the control phase where it is necessary to react fast and find a solution relatively close to the previous one, the second method is used, starting from a context established by the first method.

We would like to give a sense to the explanations these methods are capable of providing. By attaching to each constraint or task definition the reasons for defining them that way (e.g. the task "play back the recorders" was defined because there is no direct transmission during any part of the flight), we can give a set of justifications due to the context, the technology or any other consideration which tends to get forgotten as teams are renewed. Conversely, if modifications to the context appear over time, we shall be able to know the impact on solutions already found so as to be in a position to adapt and re-use them with full knowledge of the facts.

Finally, a design process being roughly an operation of finding a solution subject to constraints with re-use of the knowledge already acquired, we feel that it is possible to adapt and generalize the results of this study to the problem of designing complex systems.

6. REFERENCES

1. E.Bensana, G.Verfaillie and A.Gasquet.
Méthodologie d'élaboration et de contrôle de plan de vol pour navettes spatiales (rapport de fin de phase 1).
Technical Report 1/3393/DERI, CERT 1991.
2. E.Bensana and G.Verfaillie.
Méthodologie d'élaboration et de contrôle de plan de vol pour navettes spatiales (rapport intermédiaire de phase 2).
Technical Report 2/3393/DERI, CERT 1992.
3. T.Scheix, G.Verfaillie and B.Trousse.
Evaluation des outils de la programmation logique avec contraintes pour le traitement des problèmes à fortes combinatoire. (rapport de phase 1).
Technical Report 1/3411/DERI, CERT 1991.
4. T.Scheix, G.Verfaillie, B.Trousse and P.Charman.
Evaluation des outils de la programmation logique avec contraintes pour le traitement des problèmes à fortes combinatoire. (rapport de phase 2).
Technical Report 2/3411/DERI, CERT 1991.
5. T.Scheix, G.Verfaillie, B.Trousse and P.Berlandier.
Evaluation des outils de la programmation logique avec contraintes pour le traitement des problèmes à fortes combinatoire. (rapport de phase 2).
Technical Report 3/3411/DERI, CERT 1992.
6. U.Montanari.
Networks of constraints, fundamental properties and applications to picture processing.
Information Science, 7(2):95-132, 1974.
7. R.Dechter and J.Pearl.
Network-based heuristics for constraint satisfaction problems.
Artificial Intelligence, 34:1-38, 1988.
8. J.Gaschnig.
Performance measurement and analysis of certain search algorithms.
PhD thesis, Carnegie Mellon University, 1979.
9. A.Mackworth.
Consistency in networks of relations.
Artificial Intelligence, 8(1):99-118, 1977.
10. B.Nadel.
Constraints satisfaction algorithms.
Computer Intelligence, 5:188-224, 1989.

N94-23968

ESTIMATING THE DEEP SPACE NETWORK MODIFICATION COSTS TO PREPARE FOR FUTURE SPACE MISSIONS BY USING MAJOR COST DRIVERS

Donald S. Remer, Josef Sherif and Harry R. Buchanan
Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109

Abstract

This paper develops a cost model to do long range planning cost estimates for Deep Space Network (DSN) support of future space missions. The paper focuses on the costs required to modify and/or enhance the DSN to prepare for future space missions. The model is a function of eight major mission cost drivers and estimates both the total cost and the annual costs of a similar future space mission. The model is derived from actual cost data from three space missions: Voyager (Uranus), Voyager (Neptune), and Magellan. Estimates derived from the model are tested against actual cost data for two independent missions, Viking and Mariner Jupiter/Saturn (MJS).

Key Words: Cost Model, Tracking Network, Long-Range Planning, Cost Drivers.

1. INTRODUCTION

1.1 Project Objectives

The objective of this study is to develop a model that can be used in the early planning stages to estimate the cost to modify and/or enhance the DSN to prepare for future space mission support. Ongoing costs to operate and maintain the DSN are not included. The proposed model captures the major cost drivers of a mission such as its use of an Uplink (U/L), Downlink (D/L), Very Long Base Interferometry system (VLBI), etc. The proposed model gives cost estimates that are functions of the cost drivers and duration of a project, as demanded by its unique mission. The results of this study expand on previous cost modeling that was cost driver and mission independent. The previous work could be used in the very earliest stages of cost estimating, before the cost drivers and unique mission characteristics are defined [1,2]. The present study focuses on major cost drivers that make up the total cost of a project. In so doing, the total estimated project preparation cost will reflect only those cost drivers that pertain to that particular project, and thus a more project sensitive cost estimate is achieved. This paper is based on work previously described [3,4].

1.2 The Deep Space Network (DSN)

The National Aeronautics and Space Administration (NASA) Deep Space Network (DSN) is a multimission telecommunications and radio metric data facility used to support NASA's exploration of space, research in space science, and advanced technology investigations. The Network has facilities located in North America, Europe and Australia. The Network basic services are: (1) reception of telemetry from spacecraft; (2) transmission of commands to spacecraft; (3) measurement of radio metric data for spacecraft navigation; and (4) radio science measurements.

1.3 Overview of Paper

The purpose of this work is to develop cost models that can be used for long-range planning purposes. The first model we develop is called Model A and it helps us estimate the total cost. The second model we develop is called Model B and it helps us estimate how the total costs are distributed on an annual basis over the life of the expenditures. Both models are developed based on data from 3 projects - Voyager (Uranus), Voyager (Neptune), and Magellan. The total cost obtained from each model is then checked using data from two other independent projects, MJS and Viking. However, the time profile results from Model B are not checked with these two independent projects because the actual annual detailed data is no longer available. As future data is collected at JPL (Jet Propulsion Laboratory of the California Institute of Technology), both models will be continually checked against the actual data.

In Section 2, we define the cost drivers that are required to prepare the DSN to support future space missions. Then we summarize the methodology for collecting the cost data and the cost history. The total mission cost drivers model (Model A) is developed in this section. The model is back tested against the three missions Voyager - Uranus, Voyager - Neptune and Magellan, and an example is given to show how to use the model. In Section 3, the cost drivers time profile

model (Model B) is developed. The model is back tested against the three missions and an example is given to show how to use the model. In Section 4, as an 'external' check we compare model A to two independent projects: MJS and Viking.

2. DEVELOPMENT OF COST DRIVERS MODELS

2.1 Definition of Cost Drivers

1. M/O: Maintenance & Operations Costs - These are initial entry and management costs for the project: e.g., funding for M&O network operations, network operations project support, and the Tracking Data System (TDS) manager .
2. D/L: Downlink Frequency - These are costs of adding new receiver, antenna and microwave capability to the DSN: new downlink frequency, additional performance capability to the existing receivers, antennas and microwave, increasing the number of channels provided by the existing antenna, receivers and microwave, etc.
3. U/L: Uplink Frequency - These are costs of adding new transmitter, antenna and microwave capability to the DSN: new uplink frequency, additional performance capability to the existing transmitters, antennas and microwave, such as higher power, increased phase stability, etc.
4. TEL: Telemetry Upgrade - These are costs of upgrading the telemetry and signal processing equipment; adding new technical capability, adding to the monitor and control capability, providing new techniques such as baseband combining for antenna arrays, etc.
5. G/T: Antenna Gain/Noise Temperature - These are the costs of upgrading the ratio of antenna gain to the receiving system noise temperature (G/T). This is a figure of merit for a telecommunications receiving system. Included are costs of providing new antennas, enlarging existing antennas, providing new/improved low-noise microwave amplifiers, providing antenna arrays, etc.
6. R/M: Radio Metric Accuracy Upgrade - These are costs associated with upgrading the accuracy with which the spacecraft location can be measured. This includes upgrades to the data system equipment, improving DSN station location accuracy, improving

time synchronizing calibration throughout the network stations, etc.

7. R/S: Radio Science Upgrade - These are costs associated with upgrading the DSN radio science performance. These include adding new and/or improved receivers, data processing and recording equipment, improved frequency and timing equipment/ calibration, etc.

8. VLBI: Very Long Baseline Interferometry System (VLBI) - These are costs of implementing a new complete VLBI equipment for both 34m Wide Channel Bandwidth (WCB) and 70m Narrow Channel Bandwidth (NCB) systems. Included are receivers, low-noise amplifiers, support for Radio Source Catalog and Universal Time Engineering.

9. OTH: Other - These costs are for any miscellaneous tasks not fitting into one of the above cost driver categories. See section 2.3 for further details.

2.2 Data Collection and Summary

The annual cost obligations used in this paper are taken from Telecommunications and Data Acquisition (TDA) Work Authorization Documents (WAD Obligations Performance Reports), and do not include Construction of Facilities (CofF) costs, spacecraft costs, transportation costs and/or other logistics costs.[5] All costs used in this report are adjusted for inflation to 1987 dollars using the NASA Inflation Index. The costs for the following three projects Voyager - Uranus, Voyager - Neptune, and Magellan and the typical cost driver values are shown in Table 1. The duration of the preparation costs considered in this report are: 1982 through 1986 for Voyager (U), 1985 through 1988 for Voyager (N), and 1985 through 1988 for Magellan [1,2]. Extracting the modifications and/or enhancement cost data from the TDA Work Authorization Documents was a time consuming task. It took about one work year and numerous consultations with appropriate specialists to decide which costs were modifications and/or enhancements, and to allocate these costs to the appropriate cost drivers.

2.3 Development of the Model A: The Total Mission Cost Drivers Model

We propose that the total cost can be estimated by the summation of the typical cost driver values

Table 1. Preparation cost drivers for three missions, and typical cost-driver values, 1987 \$K^a

Cost Drivers	Missions			Typical Cost-Driver Value
	Voyager (U)	Voyager (N)	Magellan	
M/O	955	677	634	755
D/L	2,647	6,566	2,211	3,808
U/L	[700]	0	8,947	8,947
TEL	10,357	[1,134]	15,445	12,901
G/T	17,795	21,008	0	19,402
R/M	2,032	580	[33]	1,306
R/S	1,374	5,484	[46]	3,429
VLBI	0	[601]	4,436	4,436
Other	700	1,735	849	1,095
Total	35,860	36,050	32,522	56,079

^aThe bracketed numbers are discussed in Section 2.3.

given in Table 1 that are relevant to the modifications and enhancements to prepare for a future mission. The typical cost driver value in the model is an effective average value that is calculated after assigning any cost driver values that are less than 15% of the maximum to the "Other" cost driver category. We assumed that a cost value that low reflects miscellaneous changes to the system rather than a significant cost driver upgrade. For example, in Table 1, the 46 \$K value for (R/S) of Magellan is less than 15% of 5,484 \$K of Voyager (N); therefore, this cost driver for Magellan is considered "Other". Consequently, the typical (R/S) cost driver value will be the average of that of Voyager (U), and Voyager (N) or 3,429 \$K. The numbers in brackets in Table 1 were handled this way. Note that Magellan has 770 \$K of miscellaneous costs in addition to the R/M [33] and R/S [46] included in 'Other' cost driver category.

2.4 Back Testing the Total Mission Cost Drivers Model

The total mission cost drivers model (Table 1) was compared with the actuals for the three missions: Voyager (U), Voyager (N) and Magellan as shown in Table 2. A comparison of the actual cost and costs predicted by Model A for the three missions can be seen in the Totals of Table 2. The difference in predicting individual mission costs ranges from 17% below to 18.9% above actual costs for Voyager (N) and Voyager (U), respectively. The costs estimated from the model are about 1.9% below that of actual costs for Magellan.

Table 2. Actual and Model A costs for three missions, 1987 \$K

Cost Drivers	Voyager (U)		Voyager (N)		Magellan	
	Actual	Model	Actual	Model	Actual	Model
M/O	955	755	677	755	634	755
R/M	2,032	1,306	580	1,306	0	0
R/S	1,374	3,429	5,484	3,429	0	0
D/L	2,647	3,808	6,566	3,808	2,211	3,808
VLBI	0	0	0	0	4,436	4,436
U/L	0	0	0	0	8,947	8,947
TEL	10,357	12,901	0	0	15,445	12,901
G/T	17,795	19,402	21,008	19,402	0	0
Other	700	1,095	1,735	1,095	849	1,095
Total	35,860	42,696	36,050	29,795	32,522	31,942

2.5 How To Use Model A: The Total Mission Cost Drivers Model

Model A is developed from historical cost data as an average of three space missions: Voyager (U), Voyager (N), and Magellan. For example, to estimate the total cost for a mission that has the following six cost drivers: R/M, R/S, VLBI, U/L, TEL and G/T, we look up the typical cost driver values in Table 1 and sum them. A summary is given in Table 3.

Table 3. Preparation costs predicted by Model A in 1987 \$M

Cost Drivers	Cost Predicted by Model A
R/M	1.3
R/S	3.4
VLBI	4.4
U/L	8.9
TEL	12.9
G/T	19.4
Total	50.3

Model A gives us the total DSN cost to prepare for a mission; but it does not give a profile of costs over time. Now, we will look at Model B that will give the cost profile over time.

3. MODEL B: THE COST DRIVERS TIME PROFILE MODEL

3.1 Development of Cost Drivers Time Profile Model

The average annual cost for each cost driver of the three missions, Voyager (U), Voyager (N), and Magellan is calculated and shown in Table 5 as actual (A) data. The average data for each cost driver is then regressed over time and the equation that best describes the data is chosen as shown in Table 4. In each equation of Table 4, Y_t is the cost in year t , ($t = 1, 2, \dots, n$); t is the number of the year in the life of the DSN preparation cost for a mission, n is the total years of the DSN preparation; and the total cost of the DSN preparation for a cost driver is $Y(\text{Total}) = \sum Y_t$.

Table 4. The best-fit equation for each cost driver

Cost Drivers	Model	R^2
(M/O)	$Y_t = 352 - 318t + 144t^2 - 18.6t^3$	99
(D/L)	$Y_t = -3,053 + 4,558t - 1,474t^2 + 139t^3$	82
(U/L)	$Y_t = 4,561 - 25t$ (for $t = 3$ and 4)	100
(TEL)	$Y_t = -4,114 + 5,938t - 1,011t^2$	81
(G/T)	$Y_t = -2,175 + 5,880t - 1,053t^2$	99
(R/M)	$Y_t = 104 + 229t - 48.1t^2$	90
(R/S)	$Y_t = 1,339 - 2,207t + 1,218t^2 - 165t^3$	98
(VLBI)	$Y_t = -792 + 860t$ (for $t = 3$ and 4)	100

3.2 Analysis of Model B: The Cost Drivers Time Model

The equations chosen for the cost drivers are: Linear for U/L and VLBI, Quadratic for TEL, G/T, and

R/M, and Cubic for M/O, D/L, and R/S. Figures 1 and 2 show the actual costs and those predicted by the equations for typical cost drivers.

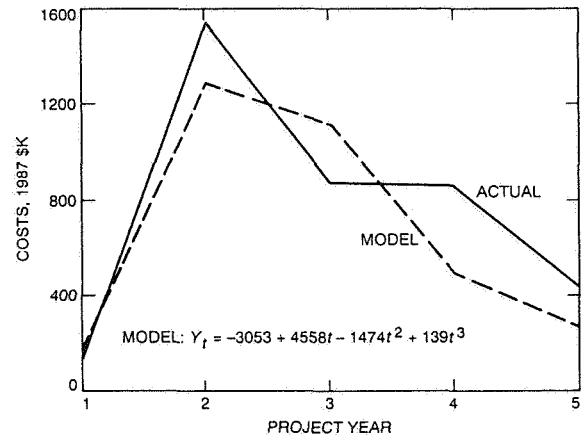


Figure 1. D/L (Downlink Frequency) Cost Driver

3.3 Back Testing Model B: The Cost Drivers Time Profile Model

Model B was checked with the three missions: Voyager (U), Voyager (N) and Magellan. Table 5 shows the average actual annual costs of the three missions as (A) data and costs predicted by the model as (M) data. Table 6 shows that the average cost per mission of 33.6 \$M as predicted by the model is 1.2 \$M below actual average cost of 34.8 \$M. The difference is about 3.4%. However, the difference in predicting total costs ranges from 22.5% below to 17.5% above actual costs to prepare for Voyager (N) and Voyager (U) respectively. The difference in predicting total cost to prepare for Magellan is 5%.

Table 5. Summary of actual average annual preparation costs for three missions and costs predicted by Model B, \$K (1987 \$). "A" = actual costs; "M" = Model B costs.

Year	M/O	D/L	U/L	TEL	G/T	R/M	R/S	VLBI	Other	Total
1 (A)	159	126	-	502	2,753	258	164	-	444	4,406
(M)	159	170	-	813	2,652	285	185	-	283	4,547
2 (A)	149	1,535	-	4,782	5,172	421	557	-	115	12,731
(M)	143	1,279	-	3,718	5,372	369	477	-	335	11,693
3 (A)	191	867	4,486	3,279	5,980	361	1,100	1,788	536	18,588
(M)	192	1,108	4,486	4,601	5,986	358	1,225	1,788	477	20,221
4 (A)	206	853	4,461	4,161	4,704	193	1,513	2,648	-	18,739
(M)	194	491	4,461	3,462	4,497	250	1,439	2,648	-	17,442
5 (A)	50	427	-	177	793	73	95	-	-	1,615
(M)	37	262	-	300	896	46	129	-	-	1,670
Total (A)	755	3,808	8,947	12,901	19,402	1,306	3,429	4,436	1,095	56,079
Total (M)	725	3,310	8,947	12,894	19,403	1,308	3,455	4,436	1,095	55,573
(M-A)	-30	-498	0	-7	1	2	26	0	0	-506

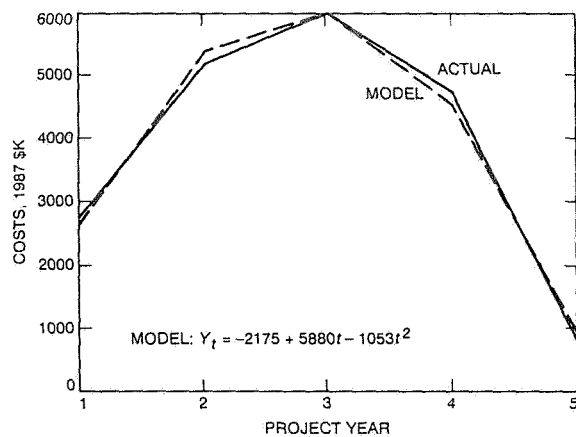


Figure 2. *G/T (Antenna Gain/Noise Temperature) Cost Driver*

3.4 How To Use Model B: The Cost Drivers Time Profile Model

As an example, to estimate the costs to prepare for five years of DSN modifications and enhancements that incur all nine cost drivers, we look up the model results in Table 5 and sum the predicted costs by the model. The results are shown in Table 7.

Table 6. Summary of the actual preparation costs for three missions, and the costs predicted by Model B, \$M (1987 \$)

Space Mission	Actual Preparation Cost, \$M	Model B Preparation Cost, \$M	Model B Minus Actual, Δ in \$M	Error, %, Δ/Actual
Voyager (U)	35.9	42.2	6.3	17.5
Voyager (N)	36.0	27.9	-8.1	-22.5
Magellan	32.5	30.8	-1.7	-5.0
Average for all missions	34.8	33.6	-1.2	-3.4

A second example is to estimate the total costs over time to prepare for five years of DSN modifications and enhancements that incur six cost drivers: R/M, R/S, VLBI, U/L, TEL and G/T. We look up the model results in Table 5 and sum the predicted costs by the model. The results are shown in Table 8.

4. EXTERNAL CHECK

4.1 Independent Missions Comparison

Model A was tested against two other independent missions: MJS and Viking. The project

Table 7. Estimate of preparation costs for a five-year, nine-cost-driver mission by Model B, \$K (1987 \$)

Year	Actual Cost-Driver Total Cost, \$K	Model B Cost-Driver Total Cost, \$K	Model Minus Actual, Δ in \$K	Error, %, Δ/Actual
1	4,406	4,547	141	3
2	12,731	11,693	-1,038	-8
3	18,588	20,221	1,633	9
4	18,739	17,442	-1,297	-7
5	1,615	1,670	55	3
Total	56,079	55,573	-506	-1

Mariner Jupiter/Saturn (MJS) incurred all major costdrivers that are covered in this paper [6]. From Table 1, the total cost for all the major cost drivers is 56.1 (1987 \$M). Based on a previous study [3], it was concluded that the MJS modification and enhancement of 10 years had two distinct phases, one for Jupiter and one for Saturn, each lasting five years. The MJS cost estimate is then twice that predicted for the five year calculation. That results in a predicted cost of 112.2 (1987 \$M) as compared to the actual cost of 97.5 (\$M) [7]. The difference is about 15%. The Viking project, which took five years of DSN modification, did not use VLBI; however, it used the other cost drivers [8]. The estimated total cost for Viking by model A is then $56,079 - 4,436 = 51,643$ (1987 \$K) or about 51.6 (\$M) as compared to the actual cost of 49.7 (\$M) [7]. The difference is 3.8%.

Model B was also tested against MJS and Viking. The total actual cost for MJS was 97.5 (\$M). The MJS cost estimate by model B is also twice that predicted for the five year calculation, as explained above and shown in Table 5. That results in a predicted cost of 2×55.573 (\$M) or 111.15 (1987 \$M). The difference is about 14%. The Viking project did not

Table 8. Estimate of preparation costs for a five-year, six-cost-driver mission by Model B, \$M (1987 \$)

Cost Drivers	Cost Predicted by Model B
R/M	1.3
R/S	3.5
VLBI	4.4
U/L	8.9
TEL	13.0
G/T	19.4
Total	50.5

use VLBI, and thus the estimated total cost for Viking by model B is $55,573 - 4,436 = 51,137$ (1987 \$K) or about 51.1 (\$M) as compared to the actual cost of 49.7 (\$M). The difference is 3%. Model B was not compared with MJS and Viking on an annual basis since the actual annual detailed data is no longer available.

5. SUMMARY

A cost model has been presented to give estimates for future DSN modification and enhancement costs to prepare for future space missions. The model has two versions: A and B.

Version A is called the total mission cost drivers model (Model A), and Version B is called the cost drivers time profile model (Model B).

Model A estimates total DSN costs based on average DSN cost drivers from three space missions: Voyager (U), Voyager (N) and Magellan. The model is concerned with those cost drivers that are relevant to a mission, and thus is sensitive to mission objectives and uniqueness. Model A does a reasonable job of representing the actual costs for Voyager (U), Voyager (N) and Magellan. Based on our back-testing the actual three projects against the model, the results are in the range of 17% below to 19% above actual costs. Model A was also compared to two other independent projects, Mariner Jupiter/Saturn (MJS) and Viking. The model gave total cost estimates which range from 15% to 4% above actual total costs for MJS and Viking respectively.

Model B estimates the annual costs of each cost driver relevant to a mission and also total mission cost. The model is time and cost driver sensitive and thus will capture future mission costs dependence on both time and relevant cost drivers. Model B was also compared to the actual DSN modification and enhancement costs over time for Voyager (U), Voyager (N) and Magellan. Based on our back testing the actual three projects against the model, the results are in the range of 22.5% below to 17.5% above actual costs. Model B was also compared to MJS and Viking. The model gave total cost estimates which range from 14% to 3% above actual total costs for MJS and Viking respectively. Both Model A and Model B are applicable to missions that do not exceed five years of duration and that have the cost drivers discussed in this study.

Acknowledgment: This research was carried out by the Jet Propulsion Laboratory, California Institute of Technology under contract with the National Aeronautics and Space Administration.

References

- [1] J.S. Sherif, D.S. Remer and H.R. Buchanan "Long-Range Planning Cost Model For Support of Future Space Missions By The Deep Space Network," TDA Progress Report 42-101, January-March 1990, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109, pp. 179-190, May 15, 1990.
- [2] D.S. Remer, J.S. Sherif, and H.R. Buchanan, "Long-Range Planning Cost Model for Support of Future Space Missions by the NASA Deep Space Network", Journal of Parametrics, 10, 1, pp 3-25 (1992).
- [3] J.S. Sherif, D.S. Remer, and H.R. Buchanan, "Modeling Preparation Costs for Space Missions by Using Major Cost Drivers," TDA Progress Report 42-106, April-June 1991, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA 91109, pp. 404-415, August 15, 1991.
- [4] Donald S. Remer, Josef S. Sherif and Harry R. Buchanan, "Modeling the Deep Space Network Costs for Future Space Missions by using Major Cost Drivers," International Journal of Production Economics, 28, 2, pp 193-202 (1992).
- [5] Obligations Performance Reports, 1982-1988," TDA Work Authorization Documents (WADs) JPL internal documents, Jet Propulsion Laboratory, Pasadena, CA 91109.
- [6] J.R. Hall, Private Communication (1990) JPL, California Institute of Technology, Pasadena, CA 91109.
- [7] J.W. Layland, Private Communication (1990) JPL, California Institute of Technology, Pasadena, CA 91109.
- [8] D.G. Mudgway, Private Communication (1990) JPL, California Institute of Technology, Pasadena, CA 91109.

KEYWORD INDEX

A

Ada, I.12
aerospace, B.9, C.3,
D.7, D.8, E.1.c, H.5
AI, C.2, L.6
AI technology, L.8
ancillary data, K.4
anomaly, C.7, C.8
AOS, G.5
AOS Testbed, G.2
APM, D.12
APM Centre, I.7
apogee maneuvers, H.6
Ariane4 test, F.7
artificial intelligence, F.1, F.8
ARQ, G.3
ASTER, A.3
astronautics, D.6
automated operations, F.2,
F.9, J.4
automatic task sequencing,
D.11
automatic tools, G.8
automation, B.10, D.10, E.1.g,
E.2.a, E.3.e, F.1, G.9, G.10,
G.12, J.1, J.5
autonomous spacecraft
operations, B.6

B

baseline, D.5

C

case-based reasoning, F.7
catalogue systems, K.12
CCSDS, B.1, B.9, G.1, G.2,
G.3, G.5, K.2
CLIPS, H.4
COES, K.8

colocation, H.10
color, I.8
Columbus, D.12, I.7
Columbus operations, E.1.d
command, D.8
command assurance, I.4
command verification, G.4
communications, I.2, K.1
computer operations, F.9
computer-human
interfaces, G.12
conflict solving, F.6
conjunction, C.7
conservatism, I.11
console automation, F.9
continuous improvement, I.3
control, D.4
control center, B.10, G.6, G.12
control system, F.11
COP-1 protocol, G.4
corporate memory, F.7
COSMICS, G.1
cost drivers, L.10
cost efficiency, L.2
cost model, L.10
crew activity planning, C.2
cross-support, H.9
customer focus, E.1.a
customer requirements, L.3
C++, K.2
C-based software, L.8

D

data acquisition, G.8
data analysis, E.3.c, K.4
data archiving, K.12
database, K.5

data compression, K.10
data monitoring, J.5
data processing, K.1
data production, K.7
data representation, I.10
data retrieval, K.6
data security, I.9
data systems, K.1, K.11
Deep Space Network,
B.2, G.10, G.11
defect prevention, I.4
design, I.11
design drivers, D.13
design process, L.9
development, L.4
digital map display, D.2
display design concepts, E.2.d
distributed architecture, G.6
distributed computing, G.6
distributed mission
operations, B.4, I.10
distributed planning, C.2
distributed systems, F.2
documentation, K.5
Doppler, C.7
DORIS, H.2
Dutch Utilisation Centre, D.4
dynamic simulation, L.1

E

Earth observation, K.12
Earth science, K.1
ECS, G.1
embedded processor, J.3
EOSDIS, A.2, G.1
ergonomics, D.11
error management, I.4
ERS-1, D.3
expert systems, D.3, F.6,
F.7, F.8, H.4, J.4, J.5, L.6
explanation, L.9

F

failure management support,
E.3.d
flexible request language, L.7
flight dynamics, H.8
flight operations, E.3.a, K.1
flight programs,
Keynote Speech (Diaz)
flight-qualifiable
components, G.2
functional analysis, L.1

G

generic functions, D.13
geometry, K.4
geostationary satellite, C.1,
H.10
geostationary satellite
positioning, H.11
GOSIP, B.2
graphical, E.3.b
graphics, A.5, E.3.e
ground blocks and
spacecraft blocks, D.5
ground control, A.1
ground control system, H.11
ground data system, A.8,
E.1.e, J.1, L.5
ground segment, Plenary
Speech (Tateno), I.7, A.7
groundtrack, D.2
groupware, F.11
GUI, F.13, H.5, I.8
GUI design style guidelines,
I.8

H
HST Servicing Mission, L.6
Hubble Space Telescope, L.6
human factors guidelines, I.8
hypermedia, E.2.a
hypertext, E.2.a

I
IGDS, G.1
information processing
 center, F.9
information retrieval, E.2.a
information systems, K.1
infrastructure, B.5
instrument swath, D.2
integration, D.4
intelligent tutoring systems,
 I.1
interference mitigation, G.13
international collaborations,
 L.5
interoperability, Plenary
 Speech (Tateno)
IOI, I.7
ISO/OSI, B.1

J
Jupiter, C.7

K
Kalman, H.6
knowledge acquisition, F.7
knowledge-based systems,
 B.3, F.6, F.8, G.10, K.5
knowledge-based systems
 V&V, H.4
knowledge engineering, G.9
knowledge representation, I.8
knowledge sharing, B.3

L
lander localization, H.7
language, I.10
lessons learned, L.2
Level 6 data, K.4
limit sensing, J.7
Logistics, I.7
long-range planning, L.10

M
Mach, G.11
Magellan, C.9, E.3.c, J.1
mainframe processing, F.9
maintenance, C.3
maneuvers, H.8
man-machine interface, D.11,
 H.8, H.11
manned space laboratory
 control center, E.1.d
Mars 94 project, H.7
Mars landers, H.7
Mars mission, B.6, H.7
memory tracking, C.9
mission control, A.1, A.7,
 B.5, C.6, E.1.b, E.2.a,
 E.3.b, F.1, F.5, J.5
mission control center, A.9,
 H.4
mission design and planning,
 B.3
mission development, E.1.c
mission information base, B.3
mission operations, Plenary
 Speech (Debatin), C.4, C.9,
 D.7, D.9, D.10, E.1.a, E.1.b,

E.1.e, E.3.c, E.3.e, I.3, I.4,
 J.4, L.2
mission planning, A.5, D.1,
 D.2, D.3, D.13
mission requirements, L.3, L.4
mission schedule, E.3.e
monitor and control, B.2,
 G.10, G.11
monitoring, D.11
multimission control, E.1.f,
 E.2.d
multimission systems,
 Plenary Speech (Debatin),
 A.8
multimedia, F.11, G.5

N
NASDA, Plenary Speech
 (Tateno)
navigation, H.5, H.9
network, B.8, G.1
network operations, B.7
NP system, F.6
nutation, C.7

O
object-oriented, Plenary
 Speech (Debatin), B.5, E.1.g,
 F.1, G.11, I.12, K.2
observation planning, K.4
onboard navigation, H.4
ontology, B.3
operability, I.2
operating organization, A.9
operational requirements,
 E.1.b
operational training, I.5
operations, A.5, A.7, D.6,
 D.13, E.1.a, E.1.f, E.1.g,
 E.2.a, F.1, G.9, G.12, H.5,
 H.11, J.1, J.6, K.5, K.11

operations concept, E.1.b
operations coordinator, D.9
operations documentation,
 C.1
operations planning, D.8
Operator Assistant, G.11, G.10
operator support, I.1
orbit determination system,
 H.2, H.3
orbital signature, J.7
organization, C.1
OSI, B.1, B.2
overlay, D.5

P
packet structure, K.8
packet telecommanding, G.4
packet type, K.8
packet utilisation, K.8
packetised telemetry, Plenary
 Speech (Debatin), E.2.b
parallel computing, D.10
parallel processing, J.2
parameter, K.8
payload operations, D.12
payload operations
 control center, A.6
Petri Nets, L.1
planetary exploration, D.6
planetary missions, B.6
planning, D.4, F.1, L.6, L.8
planning system, B.8
plotting software, E.3.c, J.2
probability, I.11
process improvement, I.3
productivity enhancement, J.4
program phases, E.1.b
prototyping, F.10, L.7

R

reactive schedule planning,
F.3
real-time, H.6, J.5
real-time operation, F.11
recovery plan, C.8
reengineering, F.1
reference model, Plenary
Speech (Fuechsel)
remote sensing systems, A.9
requirements, L.4
resource allocation planning,
L.5
resource scheduling, D.3
revision, L.9
robustness, L.9
round-trip light time, B.6
rover missions, B.6
rule-based planning, C.2

S

safing, C.8
sample return missions, B.6
satellite, I.12
satellite operations, B.8, A.9
satellites, unmanned
scientific, E.1.c
Saturn, D.6
scheduling, D.11, F.1, F.3,
F.12, G.13, I.10, L.5, L.6,
L.7, L.8, L.9
science planning, D.6
science sequencing, D.7
search and rescue, Plenary
Speech (Ivarnez)
sequencing, D.10
service and network
management, G.2

SFDU, G.5
S/I, G.13
signal-to-noise ratio, G.7
simulation, I.12
software, A.5, B.5, H.10, J.2
software reuse, L.8
SOSDU, G.2
space astronomy, D.13
space communications, B.1,
G.3, G.13, L.7
space data archives, K.6, K.7
space data system, C.4
space infrastructure, Plenary
Speech (Tateno)
space mechanics, H.11
Space Shuttle, A.1, A.5
space operations, Plenary
Speech (Fuechsel), Plenary
Speech (Ivarnez), Plenary
Speech (Tateno), I.9
Space Station, A.1, A.4,
E.3.d, K.11
Space VLBI, C.5, H.9
spacecraft, F.11, G.5, J.2,
J.7, K.5
spacecraft control center,
A.6, F.13
spacecraft control system,
E.2.b
spacecraft monitoring, E.3.b
spacecraft operations, B.3,
C.6, I.3
spacecraft payload, D.4
spacecraft tracking system,
C.4

space telecommunications
performance, G.7
spontaneous reconfiguration,
C.7
SPOT, A.9
state tracking, C.9
stationkeeping, Plenary
Speech (Ivarnez), H.10
station positioning, Plenary
Speech (Ivarnez), C.1
supervisory control, I.1
surface operations, B.6
system adaptations, A.8
system development, A.8
system engineering, I.4
system fault protection, C.8
system integration, A.8
system operations, L.1
system requirements, B.7
system testing, G.8
S/I, G.13

T

technical training, I.1
technological database, K.10
telecommand, B.1, G.3
telemetry, B.1, B.9, H.8, J.6,
J.7, K.2
telemetry processing, A.1,
J.1, J.3
teleoperations, A.4, A.6
telescience, D.12
temporal data, K.10
testbeds, B.9
testing, G.2
testing automation, G.7

testing of multimission
systems, A.8
thread, G.11
timeline, A.5
time and resources
constraints, C.2
tool, L.4
total quality, I.3
total quality management, I.4
tracing matrix, L.4
tracking network, L.10
training, I.6
transputer, J.3
trending, J.2

U

Ulysses, C.6, C.7
UNIX, H.8, J.6
unmanned scientific satellites,
E.1.c
uplink process, D.5
user interface, I.2
user operations support, E.1.d
user-configurable systems,
E.1.e

V

video conference, F.11
visualization, E.3.b, J.6
VLSI, K.2

W

workstations, B.10, J.3, J.6
worst case, I.11
X-Windows, H.5, J.5, K.2

AUTHOR INDEX

A

Acton, Jr., C. H. — K.4
 Alkalaj, L. J. — D.10
 Allard, F. — C.2
 Altunin, K. R. — C.5
 Altunin, V. I. — C.5
 Amador, A. V. — D.10
 Angelino, R. — F.9
 Angold, N. — C.6
 Arcangeli, J.-P. — K.10
 Aussenac, N. — F.7

B

Bahrani, K. A. — J.4
 Bailey, M. D. — B.7
 Bailin, S. C. — I.8
 Baize, L. — L.4
 Barnett, Y. — E.2.c
 Barbier, C. — H.6
 Barnes, V. B. — D.8
 Barr, T. H. — F.10
 Barro, E. — L.1
 Basilio, R. R. — C.8
 Bastien-Thiry, C. — L.9
 Bater, R. — G.4
 Battocchio, L. — I.7
 Beach, E. — F.14
 Benso, S. — K.6
 Benson, R. D. — B.4
 Bernard, J. — H.7
 Berthias, J.-P. — H.2
 Bliss, D. A. — D.7
 Bogovich, L. — L.6; L.8
 Bonneau, F. — H.7
 Bothwell, G. W. — A.3
 Broseghini, T. — L.8
 Brosi, F. — G.3
 Brousse, P. — H.11
 Bruno, K. J. — I.4
 Bryant, L. — E.1.a
 Buchanan, H. R. — L.10
 Bucher, A. W. — C.9; E.3.c; J.1
 Burke, G. — L.5

Burkhardt, C. — L.6; L.8
 Burliegh, S. — F.11
 Bush, J. — J.7

C

Campan, G. — H.8
 Cantin, M. R. — A.4
 Canu, C. — I.7
 Carlton, D. — E.3.b
 Carlton, M. — G.12
 Carper, R. D. — B.9
 Castro, H. — F.11
 Chafin, R. L. — E.1.f; E.2.d
 Chang, E. S. — A.2
 Childs, D. B. — K.7
 Chow, S. — F.10
 Chu, R. W. — I.1
 Ciarlo, A. — K.6
 Cipollone, G. — K.5
 Citrin, E. A. — C.3
 Cohen, O. — E.2.c
 Cooper, L. P. — B.2; G.9
 Cox, R. M. — J.2
 Crochemore, M. — K.10
 Culbert, C. — F.4
 Curran, P. S. — E.1.f; E.2.a

D

Dao, H. — L.8
 Darroy, J.-M. — F.1
 Davidson, R. A. — E.2.a
 de Boer, F. — H.6
 DeGeorges, C. — J.7
 Del Bufalo, A. — L.1
 Dell, J. — E.3.d
 Delobette, D. — H.7
 Desprairies, A. — H.11
 Dias, W. C. — B.6
 Dougherty, A. — L.6
 Dulac, J. — C.1
 Dunning, Jr., R. A. — K.11
 Durham, D. M. — C.8
 Durham, R. — L.5

E

Eckstein, M. C. — H.10
 Ellis, J. — H.9
 Ellman, A. — G.12

F

Fatig, M. — E.1.c; E.3.a
 Fayyad, K. E. — G.9
 Ferri, P. — A.7
 Finnerty, D. F. — D.6
 Fiskhind, S. A. — K.11
 Fletcher, M. — H.4
 Foley, G. T. — C.3
 Forman, M. — J.3
 Fratter, I. — D.11
 Frauenholz, R. B. — H.1
 Friedman, G. — F.11
 Fuchs, J. J. — K.6
 Fujii, S. — G.5
 Fujiwara, M. — F.6

G

Gal-Ezer, E. — E.2.c
 García-Pérez, R. — C.7
 Geller, G. N. — A.3
 Gersbach, J. — F.11
 Giancola, P. — F.14
 Gibson, S. — F.14
 Gonner, J. — H.10
 Grant, T. J. — D.4
 Green, W. B. — K.9
 Guffin, O. T. — D.13
 Gutiérrez-Luaces, B. O. — G.7

H

Ha, A. — E.3.d
 Haddow, C. R. — D.1
 Hajen, G. — C.2
 Hall, J. R. — L.3
 Hamer, P. — C.6
 Happell, N. — L.7
 Hara, H. — F.12; G.5
 Harris, J. A. — J.4

Head, N. C. — B.5
 Herman, G. — E.2.c
 Heuser, W. R. — B.2
 Hill, Jr., R. W. — G.10
 Hirata, N. — F.12
 Holder, B. E. — E.1.e
 Holmes, D. P. — L.3
 Horvath, J. C. — D.10
 Hourcastagnou, J.-N. — K.10
 Hughes, P. M. — J.5
 Hull, L. G. — I.10
 Hunter, D. G. — A.4

I

Ihara, H. — G.1

J

Jayles, C. — H.2
 Jiang, J. — I.8
 Johnson, J. — L.6; L.8
 Jones, M. — D.1
 Jowers, S. — E.3.d

K

Kahn, P. B. — B.4
 Kato, H. — G.1
 Kaufeler, J.-F. — B.5; F.5; K.8
 Kehr, J. — E.1.d
 Kellock, S. J. — A.7
 Kelly, A. C. — A.2
 Kirby, S. — E.3.d
 Kirkpatrick, R. L. — J.6
 Kispert, A. — L.6
 Klein, A. — E.2.c
 Klein, S. — L.8
 Klosko, S. — H.3
 Knops, III, F. W. — K.11
 Komatsu, S. — F.12
 Koslosky, J. — A.6
 Kosugi, S. — F.6
 Kuo, N. R. — B.7

L

Landshof, J. A. — D.9
 Larson, S. A. — A.3
 Lauderdale, J. — A.6
 Laue, H. A. — F.5
 Lauritsen, J. — E.3.d
 Ledbetter, K. W. — E.1.b; L.2
 Lee, L. — F.11; G.10
 Leibee, J. — E.3.e
 Lenefsky, Z. — E.2.c
 Leonard, Jr., R. E. — J.1
 Leppla, F. — L.5
 Levesque, M. E. — E.1.e
 Linick, S. H. — D.5
 Littlefield, R. — L.6; L.8
 Lo, M. W. — D.2
 Lockyer, P. — D.3
 Louie, J. J. — E.1.g
 Luczak, E. C. — J.5
 Lupisella, M. — E.3.e
 Lytton, D. W. — D.12

M

Macchion, D. — F.7
 Macoughtry, W. — L.3
 Madrid, G. — A.8
 Mahmot, R. — A.6; F.14
 Mandl, D. — A.6; E.3.b, J.7
 Marshall, M. H. — D.9
 Martin, D. — G.1
 Masullo, S. — I.7
 McIntyre, J. — L.8
 McKay, M. H. — K.5
 McLean, D. — L.6; L.8
 McNenny, R. — E.3.d
 Miles, Jr., R. F. — I.11
 Miller, K. J. — E.1.g
 Minnix, J. — L.7
 Mitchell, C. M. — I.1
 Mitschdörfer, P. — C.2
 Mizuta, H. — G.1
 Moe, K. L. — L.7
 Montermerlo, M. D. — F.8
 Montenbruck, O. — H.10
 Morrison, A. D. — A.3
 Müller, C. — G.4
 Muratore, J. F. — A.1
 Murphy, E. D. — I.8
 Murphy, S. C. — E.1.g

N

Newsome, P. A. — G.2
 Nichols, D. A. — A.3
 Nickum, W. — J.3
 Ninomiya, K. — C.4
 Nishihata, S. — F.12
 Noguchi, H. — B.8

O

Ochs, W. — L.6
 Ogasawara, H. — G.5
 Ohmori, M. — F.6
 Ondrus, P. — E.1.c
 Oniyama, A. — F.12
 Onken, J. F. — D.13
 Otranto, J. F. — G.2

P

Pacholczyk, P. — A.9
 Pack, G. — E.3.d
 Page, B. — L.6, L.8
 Palmer, M. D. — K.12
 Pape, U. — C.2
 Paris, J. — K.5
 Parkes, A. — K.8
 Parlier, R. — F.11
 Pasciuto, M. P. — K.11
 Pasquier, H. — L.4
 Peccia, N. M. — G.8
 Pidgeon, A. — K.8
 Pietras, J. — B.1
 Pin, J.-E. — K.10
 Poirier, P. — K.3
 Pollmeier, V. M. — H.5
 Porter, D. — L.5
 Potter, W. — L.6; L.8
 Potts, S. K. — A.5
 Potts, S. S. — I.4
 Poulter, K. J. — B.3; F.5
 Pradines, D. — H.2
 Putney, B. — H.3

Q

Quan, A. G. — F.9

R

Rackley, M. — A.6
 Rash, J. L. — G.13
 Rasmussen, A. N. — F.2
 Remer, D. S. — L.10
 Rossi, F. — L.1
 Rozenfeld, P. — I.5

S

Salazar, A. J. — B.7
 Sasaki, S. — C.4
 Scaffidi, C. — E.3.e
 Schielow, N. — C.2
 Schmitz, S. — I.9
 Schneider, K. M. — D.10
 Schulze, R. — D.12
 Schwuttke, U. M. — F.9; F.11
 Scott, C. J. — I.6
 Shendock, R. — J.7
 Sherif, J. — L.10
 Short, O. G. — E.3.c; J.1
 Smith, G. — K.1
 Smith, H. N. — B.3; F.5
 Soler, C. — F.7
 Sorensen, E. M. — D.1; G.4
 Spearing, R. — L.3
 Spitale, J. N. — D.10
 Sukhanov, K. G. — C.5
 Sweetin, M. — I.3

T

Takahashi, T. — B.8
 Tanaka, K. — B.8
 Tavernier, G. — H.8
 Theis, G. — B.1
 Thorn, K. E. — K.2
 Troendly, G. — J.3
 Truong, T. — E.3.d
 Truszkowski, W. F. — I.8
 Tuchman, A. — L.6; L.8
 Tumura, K. — G.5

U

Uljon, L. J. — A.1
 Urista, J. — G.11

V

Vaules, Jr., D. — E.3.b
 Veregge, J. R. — F.9
 Verfaillie, G. — L.9
 Viallefont, P. — G.6
 Vo, D.-P. — F.7

W

Wall, S. D. — E.1.b; L.2
 Walzer, B. — E.2.c
 Wanczuk, G. — A.8
 Wang, L. — H.4
 Weaver, S. — J.7
 Weld, K. R. — D.5
 Welz, L. L. — I.4
 Wessen, R. R. — D.6
 Wheadon, J. — E.2.b; F.3
 Wilkinson, B. — I.2
 Wilkinson, B. M. — B.7
 Williams, A. P. — I.12
 Williams, J. M. — K.12
 Witkowski, M. M. — I.4
 Wojcik, Z. A. — A.4
 Wolff, T. — D.1
 Wong, Y. F. — G.13

Y

Yamada, S. — F.12; G.5
 Yamada, T. — C.4
 Yamamoto, Y. — F.12; G.5
 Yamaya, K. — B.8; F.6
 Yambe, M. — F.6
 Yen, W. — L.8
 Yokomizo, M. — G.5
 Yoshida, F. — B.8

Z

Zaouche, G. — B.10
 Zelensky, N. — H.3
 Zweben, M. — F.13

ACKNOWLEDGMENTS

Several years ago, ESOC recognized that no single forum existed for the international space mission operations community. To encourage the development of such a forum, ESOC convened the First International Symposium on Ground Data Systems for Spacecraft Control, held in June 1990 in Darmstadt, Germany. The success of that symposium inspired JPL to plan and host SPACEOPS 92. The SPACEOPS 92 Management Committee

would like to acknowledge and thank ESOC for initiating this event. These JPL organizations provided particular support in planning the symposium: Office of Flight Projects, Office of Space Science and Instruments, Office of Technology and Applications Programs, and Office of Telecommunications and Data Acquisition. In addition, the following individuals contributed their time and energies to make SPACEOPS 92 a success:

Cindy Alarcan-Rivera	Susan Duca	Joseph Johnson	Paul Ondrus
Scott Alexander	David Eisenman	Lino Jubilado	Jimmie Overstreet
Karen Robinett Altunin	Robert A. Erwin	Christopher Kelly	William Pateracki
Valery Altunin	James A. Evans	Elsa King	Duane Patterson
Shawn Austin	Dana Flora-Adams	Takashi Kiriya	Corinne Rice
Richmond Benton	Maureen Flynn	Carol Lachata	Veronika Riedel
Debra Bloomfield	Michael Forman	Judy Levstik	Audrey Riethle
Jan Bostater	Helen Frawley	Kim Lievense	Maxine Riffel
Boualim Bousseloub	Robert Garcia	John Louie	Tom Ryan
Scott Bowdan	David Gardner	Pamela Malone	David Seidel
Tim Brice	James Gersbach	Debbie Martin	James Shaw
Geraldine Bridges	Nancy Gifford	John Matlock	Joseph A. Smith, Jr.
Thomas Bunker	Bobbie Grable	John Mattson	Judith P. Smith
Madeline Butler	Ana Maria Guerrero	Constance McCaig	Fred Tomey
Duane Bygum	James Guglielmino	Judith McGavin	Bob Turco
David Carlson	Peter Hanegraaf	Beverly Mendoza	Nicole VanderHorst
Henry Castro	Jennifer Harris	Sanjoy Moorthy	Kay Van Lepp
Judy Conklin	Michael Hughes	Marilyn Morgan	Layne Whyman
Azucena Costantino	Craig Hutchings	Richard Moulder	Bob Wilson
Robert Curiel	Adriane Jach	Cassie Mulnix	Penelope Wolfe
Patrick Curran	Gwen Jackson	Susan Murphy	Harry Woo
Roger Davidson	Sheryl Jackson	Helga Mycroft	Michael Workman
Mark Del Mar	Michael Jahanshahi	James Nations	Robyn Young
Elisabeth Dettinger	Kaaren Jensen	Michael Nieto	

SYMPOSIUM EXECUTIVE COMMITTEE

Norman R. Haynes
Chairperson

Richard C. Coffin
Paper Selection

Patricia B. McLane
Logistics/Registration

Garry M. Burdick
Vice Chairperson

Esker K. Davis
Program

Ursula M. Schwuttke
Posters and Demonstrations

Judith A. Cobb
Manager

Dorothe L. Horttor
Publications

Richard J. Southern
Finance

Gene Burke
Mailing List/Database

Frank D. Kuykendall
Technical Tours

Giulio Varsi
International Relations

ESA REPRESENTATIVES

Kurt Debatin

Carlo Mazza

SYMPOSIUM ADVISORS

Peter Beech
*European Space
Operations Centre*

Merle McKenzie
Jet Propulsion Laboratory

John C. Rodgers
*NASA Office of
Space Operations*

Anthony R. Gross
NASA Ames Research Center

Edward L. McKinley
Jet Propulsion Laboratory

Gray L. Settle
*NASA Marshall Space
Flight Center*

Robert K. Holkan
NASA Johnson Space Center

William I. McLaughlin
Jet Propulsion Laboratory

Guenther K. Strobel
*NASA Office of Space Science
and Applications*

William N. Jensen
Jet Propulsion Laboratory

David W. Moxley
NASA Kennedy Space Center

Maurice Winterholer
*Centre National d'Études
Spatiales*

James F. Jordan
Jet Propulsion Laboratory

Helen N. Paley
Jet Propulsion Laboratory

Arthur I. Zygielbaum
Jet Propulsion Laboratory

T. David Linick
Jet Propulsion Laboratory

Dorothy C. Perkins
*NASA Goddard Space
Flight Center*



National Aeronautics and
Space Administration

Jet Propulsion Laboratory
California Institute of Technology
Pasadena, California